DISSERTATION

Reasoning about Specifications in Model Checking

ausgeführt zum Zwecke der Erlangung
des akademischen Grades eines
Doktors der technischen Wissenschaften

unter der Leitung von
Univ.Prof. Dipl.-Ing. Dr.techn. Helmut Veith
Institut für Informationssysteme (184/2)

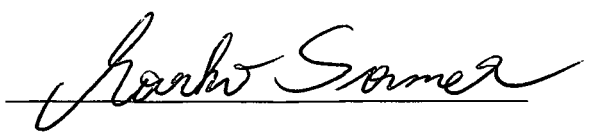
eingereicht an der
Technischen Universität Wien
Fakultät für Informatik

von

Marko Samer

9 7 2 5 3 6 5
A-7532 Litzelsdorf 336

Wien, im September 2004



Kurzfassung

Eine der praktisch erfolgreichsten formalen Verifikationstechniken ist *Model Checking*, ein Ansatz um logische Eigenschaften endlicher Zustandssysteme algorithmisch zu verifizieren. Auf Eingabe eines Systemmodells in Form eines endlichen Zustandsübergangsgraphen und einer in einer Temporallogik formulierten Spezifikation, stellt ein Model Checker automatisch fest, ob das Modell die Spezifikation erfüllt. In dieser Dissertation werden verschiedene Ansätze zum Analysieren von Spezifikationen im Allgemeinen und temporallogischen Spezifikationen im Speziellen untersucht. Durch derartige Methoden können zusätzliche Informationen gewonnen werden, um zu entscheiden, ob das zu untersuchende System seine Anforderungen erfüllt. Insbesondere werden die folgenden Fragen behandelt:

(i) Was kann über die Zusammensetzung zweier Teilsysteme, die ihre jeweiligen Spezifikationen erfüllen, ausgesagt werden? Diese Frage ist besonders dann von Bedeutung, wenn ein zusammengesetztes System zu groß ist um als Ganzes untersucht zu werden. Der Schwerpunkt der vorliegenden Arbeit liegt bei Systemen deren Komponenten zirkulär voneinander abhängen. In diesem Kontext wird eine abstrakte *zirkuläre Schnittregel* und eine abstrakte *zirkuläre kompositionelle Schlussregel* präsentiert.

(ii) Wie kann eine unvollständige temporallogische Spezifikation effizient vervollständigt werden, so dass sie von einem gegebenen Modell erfüllt wird? Derartige unvollständige Spezifikationen werden *temporallogische Queries* genannt. Der Schwerpunkt der vorliegenden Arbeit liegt bei Queries mit einer einzigen stärksten Lösung. Unter anderem werden syntaktische Fragmente solcher Queries und effiziente Lösungsalgorithmen präsentiert.

(iii) Wenn ein gegebenes Modell seine Spezifikation erfüllt, erfolgt dies auf die intendierte Art? Die Beantwortung dieser Frage ist in der Literatur bekannt unter dem Namen *Vacuity Detection*. Sie ist praktisch von großer Bedeutung, da vacuous (d.h., auf triviale Weise) erfüllte Spezifikationen oft auf ein Problem im Systemdesign oder der Spezifikation hinweisen. In der vorliegenden Arbeit wird der klassische Vacuity-Begriff durch eine Parametrisierung verallgemeinert, die es ermöglicht ein Problem zu lösen, auf das von Amir Pnueli hingewiesen wurde.

Abstract

One of the practically most successful formal verification techniques is *model checking*, an approach for algorithmically verifying logical properties of finite state systems. Given a system model in the form of a finite state transition graph and a specification expressed in some temporal logic, a model checker automatically determines whether the model satisfies the specification. In this thesis, we investigate several approaches in order to reason about specifications in general and temporal logic specifications in particular. By such reasoning methods, it is possible to obtain additional information to support verification and validation engineers in their task to decide whether the system under consideration satisfies its requirements. In particular, we are interested in three main questions:

(i) Given the specifications satisfied by two systems, what can we say about the system obtained by composing these systems? This question is of great importance for composed systems that are too large to be handled at once. Our focus lies in systems whose components depend on each other in a circular manner. In this context, we present an abstract *circular cut* and an abstract *circular compositional reasoning* rule.

(ii) Given a system model and an incomplete temporal logic specification, how can the given specification be efficiently completed such that it is satisfied by the model? Incomplete specifications of this kind are called *temporal logic queries*. Our focus lies in queries with single strongest solutions. Among other things, we present syntactic fragments of such queries and efficient algorithms for solving them.

(iii) Given a system model that satisfies its specifications, does the model satisfy the specification in the intended way? In the literature, deciding this question is well known under the name *vacuity detection*. It is of great importance in practice, since vacuously (i.e., due to a trivial reason) satisfied specifications often point to a real problem in either the system design or the specification. We generalize the classical notion of vacuity by a parameterization, which enables us to solve a problem posed by Amir Pnueli.

Danksagung

Ich möchte Prof. Helmut Veith für die Betreuung dieser Dissertation sowie für hilfreiche Vorschläge und Kommentare danken. Weiters möchte ich Prof. Georg Gottlob danken, der sich bereit erklärt hat, als Zweitgutachter zu fungieren. Schließlich gilt mein besonderer Dank meinen Eltern, Theresia und Richard Samer, für ihre langjährige Unterstützung, durch die sie mir ermöglicht haben, mich auf mein Studium zu konzentrieren.

Acknowledgments

I would like to thank Prof. Helmut Veith for supervising this thesis as well as for useful suggestions and comments. Further, I would like to thank Prof. Georg Gottlob for agreeing to act as second reader. Finally, my special gratitude goes to my parents, Theresia and Richard Samer, for their support over many years by which they enabled me to concentrate on my studies.

Contents

Chapter 1

Introduction

1.1 Motivation and Aims

It is an indisputable fact that our daily life is becoming more and more dependent on automated systems. Especially in the last decades much research and technological developments in this area have been made and no end of this tendency is in sight. For example, control systems for air, railway, and road traffic, medical instruments, weapon systems, space flight systems, nuclear power control systems, financial systems, telecommunication systems, etc. are nowadays at least partially realized by integrated computer systems. Since failures of such critical systems may result in endangerment of human life or high costs, it is a very important and challenging problem to ensure their reliability.

In this thesis, we are concerned with the following tasks (IEEE Std 1012-1998) within the extensive discipline of *quality assurance* in order to – at least partially – ensure system reliability:

- *Verification*
  Confirmation by examination and provisions of objective evidence that specified requirements have been fulfilled.

- *Validation*
  Confirmation by examination and provisions of objective evidence that the particular requirements for a specific intended use are fulfilled.

The difference between verification and validation [WF02] is that verification is based on a specification whereas validation is based on the intended use of the system under consideration. Since the latter one requires the

validator to *understand* the problem domain, it is impossible to completely formalize the validation process and make it fully automatically. In contrast, verification means to demonstrate that the system is consistent with its specification [CS00, LC00]. This can be done by mathematical methods in order to guarantee a maximal degree of objective evidence. If both the specification and the system are given as formal expression and formal model respectively, the verification process can in principle be performed fully automatically. Such a formalization allows to detect very subtle defects in the logic of the system that are unlikely to be found by other quality assurance measures (e.g., testing).

Note, however, that even formal verification cannot guarantee the correctness of a system, since the specification or the system model may be faulty. Formal verification only increases the confidence in the system correctness.

One of the practically most successful formal verification techniques is *model checking*, an approach for algorithmically verifying logical properties of the behavior of finite state systems [McM00]. In particular, given a system model in the form of a finite state transition graph and a specification expressed in some *temporal logic* (i.e., a logic formalism that is well suited for specifying the relationship of events in time), a model checker automatically determines whether the system model satisfies the specification. In the negative case, it supplies a counterexample, i.e., a behavioral trace that shows that the specification is not satisfied by the model.

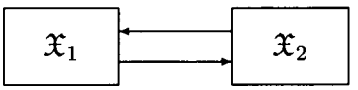
Model checking was invented in the early 1980s and has been successfully applied to the verification of computer hardware designs. For example, by using model checking it was possible to find a number of previously undetected errors in the IEEE Futurebus+ cache coherence protocol (IEEE Std 896.1-1991) [CGH+95]. Due to the success of model checking in hardware verification, there has also been much research in the last years for applying model checking in protocol and software verification.

As mentioned above, a central notion in the context of verification and validation is the one of a *specification*, i.e., a statement of requirements for a system. Specifications are also important in other phases of system development [Win00]: In requirement analysis, it helps to crystallize the customer's possibly vague ideas and reveals contradictions, ambiguities, and incompleteness in the requirements. In system design, it captures precisely the interfaces between the system components. In verification, it is the statement against which the system is proven correct. In validation, it can

be used to generate test cases for black-box testing. Finally, it serves as a kind of documentation since it is an alternative, usually more abstract, description of the system behavior.

In this thesis, we investigate several approaches in order to reason about specifications in general and temporal logic specifications in particular. Since specifications are meant to describe the system behavior as precisely as possible, reasoning about specifications provides additional information to support verification and validation engineers in their task to decide whether the system under consideration satisfies the specified and intended requirements. In particular, we are interested in three main questions:

1. Given the specifications satisfied by two systems $\mathfrak{X}_1$ and $\mathfrak{X}_2$, what can we say about the system obtained by composing $\mathfrak{X}_1$ and $\mathfrak{X}_2$? This question is of great importance for composed systems that are too large to be handled at once. If $\mathfrak{X}_2$ depends on $\mathfrak{X}_1$ but not vice versa, then the specification satisfied by the sequential composition $\mathfrak{X}_1;\mathfrak{X}_2$ can be easily obtained. However, a much more challenging problem is the case of the parallel composition $\mathfrak{X}_1\|\mathfrak{X}_2$ of circular dependent systems, i.e., $\mathfrak{X}_1$ depends on $\mathfrak{X}_2$ and vice versa:

   Since $\mathfrak{X}_1$ and $\mathfrak{X}_2$ depend on each other, specifications are in general satisfied by $\mathfrak{X}_1$ only under assumptions on $\mathfrak{X}_2$ and vice versa. Their circular composition, however, does not satisfy both their specifications (without assumptions) in general, i.e., specifications of the composed system cannot be concluded in a straightforward way from the specifications of its components. In the literature, several composition rules for specific formalisms and restricted classes of assumptions and specifications that allow this kind of reasoning were presented.
   In this thesis, we are interested in an abstract consideration of such circular compositions, i.e., on abstract composition rules that allow us to conclude specifications satisfied by a composed system from specifications satisfied by its circular dependent components independently

of a particular formalism. Within this context, our focus lies on circular compositions of proofs, i.e., $\mathfrak{X}_1$ and $\mathfrak{X}_2$ are considered as proofs and the corresponding specifications are considered as intermediate results that were proved under certain assumptions. Then, by circular composition rules, the intermediate results can be combined in order to obtain stronger results that hold without assumptions. The cut rule in logic calculi can be seen as sequential counterpart of such a circular composition rule under this interpretation.

2. Given a system model and an incomplete temporal logic specification, how can the given specification be efficiently completed such that it is satisfied by the model? Incomplete specifications of this kind are called *temporal logic queries*, that are temporal logic formulas with variables that have to be instantiated. In this terminology, our reasoning task is to solve temporal logic queries in a given model. More formally, given a model $\mathfrak{M}$ and a temporal logic query γ, we want to find all formulas φ such that $\mathfrak{M} \models \gamma[\varphi]$, that is, $\mathfrak{M}$ satisfies the specification resulting from replacing the variables in γ by φ.
This kind of temporal logic queries was first considered by William Chan [Cha00]. He presented a syntactic fragment for which he claimed that all queries in this fragment have a single strongest solution in all models. Moreover, he presented an efficient algorithm based on this property for solving queries in his fragment.
In this thesis, we are interested in general properties and syntactic fragments of temporal logic queries with single strongest solutions following Chan. More formally, we are interested in temporal logic queries γ satisfying that for every model $\mathfrak{M}$ in which γ has a solution, there exists a solution ξ such that:

$$\mathfrak{M} \models \gamma[\varphi] \quad \text{if and only if} \quad \xi \Rightarrow \varphi$$

Several properties of temporal logic queries are scattered through the literature; we are going to systematically investigate them and their relationships. Moreover, we are interested in syntactic characterizations of temporal logic queries with strongest solutions for the most important temporal logics LTL and CTL, as well as in algorithms for solving such queries. Note that Chan presented an efficient algorithm

for his fragment of CTL queries. However, in consideration of the fact that his fragment was shown to be erroneous and that he neither proved the correctness nor did he describe the underlying intuition and mathematical principles of his algorithm, it is uncertain whether the algorithm is correct. Thus, we are going to investigate Chan's algorithm. In particular, we want to find out how Chan's algorithm works and how it is related to the properties of queries in his fragment. In addition, we want to prove the correctness of the algorithm – provided that it is correct – and we are interested in generalizations of Chan's and other existing temporal logic query solving algorithms.

3. Given a system model that satisfies its specification, does the model satisfy the specification in the intended way? In the literature, deciding this question is well known under the name *vacuity detection.* Vacuously (i.e., due to a trivial reason) satisfied specifications often point to a real problem in either the system design or the specification, i.e., either the system does not work in the intended way or the specification does not describe the system behavior in the intended way. Since the intention of the system designer is crucial in this context, vacuity detection obviously belongs to the field of validation.
There exist several approaches in the literature for detecting vacuity. One of the most important ones is the work of Orna Kupferman and Moshe Vardi [KV99]. Essentially, they showed that in many interesting cases vacuity detection of a specification φ can be reduced to model checking formulas $\varphi[\psi \leftarrow \bot]$, i.e., formulas obtained by replacing a subformula ψ of φ by the constant truth value $\bot$.
Note that vacuity detection in this sense is closely related to queries: When replacing a subformula ψ by a variable instead of the constant truth value, we obtain a query γ. Then, we have

$$\varphi \text{ holds vacuously in } \mathfrak{M} \quad \text{if and only if} \quad \mathfrak{M} \models \gamma[\bot]$$

In this thesis, we are interested in the consequences of this simple observation and are going to thoroughly investigate the resulting generalized framework. In particular, based on our insights we want to solve a problem posed by Amir Pnueli [Pnu97], where he presented an example of a vacuously satisfied specification that does not fall under the common notion of vacuity.

1.2 Results

In this section, we summarize the results achieved in the scope of this thesis:

1. **A Circular Cut Rule**
   We present a circular counterpart of the classical *cut rule* in logic calculi. With our *circular cut rule*, it is possible to compose proofs that depend on each other in a circular manner. We could not find such a proof-theoretic inference rule in the literature, although there exist several approaches on circular compositional reasoning. The idea formalized in our circular cut rule is used in our proofs of circular dependent LTL and CTL query languages (cf. items 6 and 7 below).

2. **A Circular Composition Rule**
   In analogy to the circular cut rule, we present a *circular composition rule* that formalizes circular compositional reasoning on a very abstract level. In particular, our inference rule formalizes the common idea of various approaches on circular compositional reasoning in the literature. With this rule at hand, arbitrary circular dependent specifications of a certain form can be composed.

3. **Basic Relationships**
   Motivated by Chan's temporal logic queries that are guaranteed to have a single strongest solution in every model, we investigate several properties and their relationships. In particular, we study *exact* queries, i.e., queries that always have a solution that exactly characterizes the set of all solutions if there exists any solution. We show that a query is *exact* if and only if it is *distributive over conjunction* if and only if it is *monotonic* and *collecting*. Although such relations were already mentioned in the literature, this is the first time that they are thoroughly investigated and proved in this generality. Moreover, we show that the notion of monotonicity as introduced for temporal logic queries is strictly weaker than the notion of monotonicity used in database theory. These results were partially published in [SV04b].

4. **Complexity Results**
   We fix the shortcomings in Chan's proof that deciding exactness of CTL queries is EXPTIME-complete. In analogy to this proof, we show that deciding exactness of LTL queries is PSPACE-complete.

5. **Exact LTL Queries**
   We present a syntactic characterization of exact LTL queries. In particular, we define a context-free template grammar capturing all monotonic single-variable LTL queries. Then, the non-terminals in this grammar are divided into two classes. We prove that:

   - All instantiations of templates derived from the non-terminals in the first class are exact.
   - For all templates derived from the non-terminals in the second class there exists a simple instantiation that is not exact.

   Both results are obtained by nested inductive proofs with complex circular dependencies (cf. item 1). Note that such a template characterization does not contradict the PSPACE-completeness of deciding exactness of LTL queries. These results were published in [SV04b].

6. **Exact CTL Queries**
   We present a large syntactic fragment of exact CTL queries. In particular, we define a context-free template grammar such that all instantiations of templates derived in this grammar are exact. This result is obtained by nested inductive proofs with complex circular dependencies (cf. item 1). Unfortunately, we are not able to characterize exact CTL queries as in the case of LTL. However, we argue that such a characterization for CTL is a much more difficult task.
   Note that the fragment of exact CTL queries presented here is much more extensive than the fragment presented in the author's diploma thesis [Sam02, SV03]. Also the proofs are much more complex.

7. **Algorithmic Aspects of Exactness**
   Motivated by Chan's algorithm for solving exact CTL queries that are guaranteed to have a solution in every model, we thoroughly investigate those properties of such queries that are exploited in the algorithm. In particular, we show that non-determinism can be eliminated when solving exact queries is reduced to solving their subqueries. The property that enables this is called *intermediate collecting*, an auxiliary property in our proofs of exactness.

8. **Correctness of the Chan Algorithm**
   Based on our insights described in item 7, we prove that the Chan

algorithm is correct when applied to the subclass of exact CTL queries in our syntactic fragment (cf. item 6) that are guaranteed to have a solution in every model.

9. **Extension of the Chan Algorithm**
   We present a generalization of Chan's algorithm for solving all queries in our syntactic fragment of exact CTL queries (cf. item 6). In addition, we prove the correctness of this extended Chan algorithm.

10. **Non-propositional Solutions**
    Several algorithms for solving temporal logic queries have been developed in order to compute propositional solutions. We show how these algorithms can be modified in order to compute also non-propositional solutions. These results were published in [SV04a].

11. **Parameterized Vacuity**
    We show how the classical notion of vacuity can be redefined in terms of temporal logic queries and how vacuity detection can be reduced to query solving. This provides us with a new point of view of vacuity detection that leads naturally to a generalization by parameterizing the vacuity detection process with a partially ordered set of *vacuity causes*. Classical vacuity corresponds then to a parameterization with the constant truth values. We call our generalized form of vacuity *weak vacuity*. These results were published in [SV04a].

12. **Problem posed by Amir Pnueli**
    We demonstrate that a problem posed by Amir Pnueli can be easily solved by our generalized notion of vacuity. In particular, Pnueli presented an example of a vacuously satisfied specification that does not fall under the common notion of vacuity. However, it falls under our notion of weak vacuity. This result was published in [SV04a].

1.3 Overview

This thesis is organized as follows: In Chapter 2, we summarize the background knowledge necessary to understand the contents of the following chapters. In particular, in Section 2.2, we introduce Kripke structures.

Then, in Section 2.3, we introduce the most important temporal logics LTL and CTL followed by a short overview of other important temporal logics. Moreover, in Section 2.4, we describe model checking in general and symbolic model checking in particular. Afterwards, in Section 2.5, we introduce temporal logic queries and additional temporal operators used in the context of temporal logic queries.

In Chapter 3, we explore the composition of circular dependent systems. In particular, in Section 3.2, we present a circular counterpart of the classical cut rule in logic calculi. Then, in Section 3.3, we show how the circular cut rule can be applied in mutual inductive proofs. Moreover, in Section 3.4, we deal with sequential and circular compositional reasoning rules. Afterwards, we give an overview of related work in Section 3.5.

In Chapter 4, we investigate exact temporal logic queries. In particular, in Section 4.2, we consider several properties of temporal logic queries and prove their relationships. Then, in Section 4.3, we present our syntactic characterization of exact LTL queries together with the corresponding proofs. Afterwards, in Section 4.4, we present our syntactic fragment of exact CTL queries together with the corresponding proofs.

In Chapter 5, we analyze algorithms for solving temporal logic queries. In particular, in Section 5.2, we thoroughly investigate those properties of queries that are exploited in Chan's algorithm, we prove the correctness of Chan's algorithm, and we generalize Chan's algorithm. Moreover, in Section 5.3, we show how other temporal logic query solving algorithms can be extended in order to compute also non-propositional solutions.

In Chapter 6, we deal with the task of detecting vacuously satisfied specifications. In particular, in Section 6.2, we recall the basic notions of vacuity detection. Then, in Section 6.3, we generalize classical vacuity to weak vacuity and demonstrate its usefulness by several examples. Afterwards, we give an overview of related work in Section 6.4.

Finally, we conclude in Chapter 7. In particular, we summarize the most important results and we point out some remaining open questions.

Chapter 2

Background and Basic Notions

2.1 Introduction

In this chapter, we introduce the formal framework of this thesis. In particular, we consider the technological background and mathematical formalisms of *model checking*, a formal verification technique invented in the early 1980s by Clarke and Emerson [CE82, EC82] and Queille and Sifakis [QS82]. Given a model of the system to be verified and the corresponding specification, model checking is to algorithmically decide whether the model satisfies the specification. The formalisms used to build the system models are *Kripke structures* and to express the specifications are *temporal logics*. The most important temporal logics in the context of model checking and also in the context of this thesis are the *linear temporal logic LTL* and the *computation tree logic CTL*.

In addition, we introduce *temporal logic queries*, a formalism that allows to extend model checking in such a way that specifications may contain second-order variables that have to be instantiated appropriately. Temporal logic queries are a basic concept used throughout this thesis which enables us to reason about specifications in model checking in a uniform way. The text in this chapter is complemented by many references to which we refer the interested reader for further information on these topics.

This chapter is organized as follows: In Section 2.2, we introduce Kripke structures and related terms. Afterwards, in Section 2.3, we define the most important temporal logics in model checking. Model checking in general and symbolic model checking in particular are then described in Section 2.4. Afterwards, Section 2.5 deals with temporal logic queries and a syntactic extension of temporal logics. Finally, we summarize in Section 2.6.

2.2 Kripke Structures

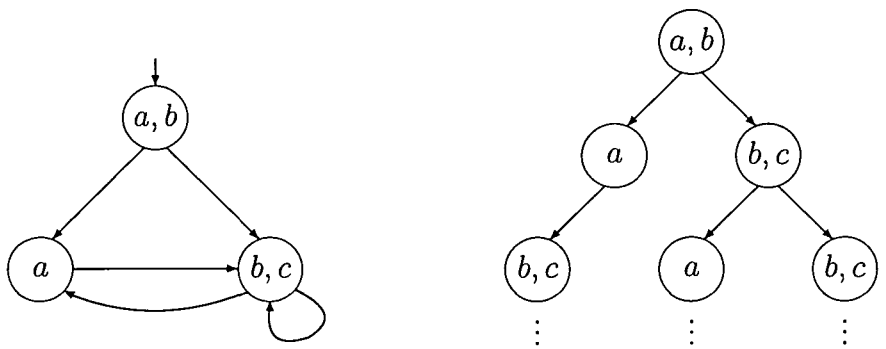
Kripke structures are named after the philosopher and logician Saul Kripke who published – among other things – fundamental works on the semantics of modal logics [Kri63a, Kri63b]. His semantic approach is known by the names *possible world semantics*, *relational semantics*, and *Kripke semantics*. The key idea is to evaluate modal logic formulas in such a way that the modal operators switch between possible worlds, each of them having their own valuation of atoms. More precisely, modal logic formulas are evaluated over Kripke models consisting of a set of possible worlds, an accessibility relation between these worlds, and a valuation function assigning to each atomic proposition the set of possible worlds at which the atom evaluates to true. In this generalized framework, classical logic amounts to the special case of a single possible world.

The interested reader is referred to Bull and Segerberg [BS84], Stirling [Sti93], Hughes and Cresswell [HC98], Blackburn et al. [BdRV01], and Clarke and Schlingloff [CS01] for an exhaustive introduction.

In model checking, Kripke models are usually called Kripke structures, the possible worlds are called states, the accessibility relation is called transition relation, and the valuation function is usually considered as labeling function that labels each state with a set of atomic propositions. This different terminology reflects the the more technological point of view in model checking, where Kripke structures are used to model systems that have to be verified. In particular, the states together with their labeling model the system states at certain points in time and the transition relation models the system behavior over time. A Kripke structure can therefore be seen as a non-deterministic finite state machine without input relation whose states are labeled with atomic propositions.

Definition 2.1 (Kripke structure). A *Kripke structure* $\mathfrak{K}$ over a set of atomic propositions $\mathcal{A}$ is a tuple $\mathfrak{K} = (\mathcal{Q}, \Delta, s_0, \ell)$, where

- $\mathcal{Q}$ is a non-empty and finite set of states.
- $\Delta \subseteq \mathcal{Q} \times \mathcal{Q}$ is a total transition relation.
- $s_0 \in \mathcal{Q}$ is the initial state.
- $\ell : \mathcal{Q} \to \wp(\mathcal{A})$ is a total labeling function.



Kripke structure *Computation tree*

Figure 2.1: A Kripke structure and its computation tree

For complexity considerations we denote the size of a Kripke structure $\mathfrak{K}$ by $|\mathfrak{K}|$, which is defined as $|\mathcal{Q}| + |\Delta|$. It is easy to see that, starting at any state, a Kripke structure can be unwound into an infinite tree, the *computation tree*. Note that the computation tree contains no leaves because of the totality of the transition relation. Figure 2.1 shows the graphical representation of a Kripke structure and its corresponding computation tree after unwinding the structure at the initial state.

Alternatively to computation trees, Kripke structures can also be seen as a representation of the execution sequences starting at each state, i.e., the set of branches of the corresponding computation tree. Such execution sequences are called *computation paths.*

Definition 2.2 (Computation path). A *computation path* or simply a *path* π in a Kripke structure $\mathfrak{K} = (\mathcal{Q}, \Delta, s_0, \ell)$ is an infinite sequence of states $\pi : \mathbb{N} \to \mathcal{Q}$ such that $(\pi(i), \pi(i+1)) \in \Delta$ for all $i \in \mathbb{N}$. We write π^n to denote the computation path suffix of the computation path π satisfying $\pi^n(i) = \pi(n+i)$ for all $i \in \mathbb{N}$.

For any state s in a Kripke structure $\mathfrak{K}$, we write $\mathrm{paths}(s)$ to denote the set of computation paths in $\mathfrak{K}$ satisfying $\pi(0) = s$ for all $\pi \in \mathrm{paths}(s)$. Moreover, for any set of states $\mathcal{Q}$, we write $\mathrm{paths}(\mathcal{Q})$ to denote $\bigcup_{s \in \mathcal{Q}} \mathrm{paths}(s)$. For any atomic proposition α and set of states $\mathcal{Q}$, we write $\alpha \in \ell(\mathcal{Q})$ and $\alpha \notin \ell(\mathcal{Q})$ to denote $\alpha \in \ell(s)$ and $\alpha \notin \ell(s)$ respectively for all $s \in \mathcal{Q}$.

If we use a set of paths Π where a set of states is expected, then Π denotes the set of states $\{\pi(0) \mid \pi \in \Pi\}$. Given a set $\mathcal{I} \subseteq \mathbb{N}$ of indices, we write $\pi^{\mathcal{I}}$ to denote the set of paths defined by $\pi^{\mathcal{I}} = \{\pi^i \mid i \in \mathcal{I}\}$. Note that we will mostly use interval notation for the index set $\mathcal{I}$.

2.3 Temporal Logics

Temporal logics are logic formalisms for expressing temporal properties. Although such properties can also be expressed in classical higher-order logic, temporal logics are more intuitive since they hide the explicit time variables in temporal modalities like "eventually" or "always".

The philosopher and logician Arthur Prior [Pri57] is considered the founding father of modern temporal logic. His modal logic based approach is also known as *tense logic*. The applicability of temporal logics for describing system properties and therefore its usefulness in formal verification was discovered in the 1970s. One of the pioneers in this area was Amir Pnueli with his seminal work on program verification [Pnu77, Pnu81].

The interested reader is referred to Emerson [Eme90], Manna and Pnueli [MP92], Stirling [Sti93, Sti01], Gabbay et al. [GHR94, GRF00], Øhrstrøm and Hasle [ØH95], van Benthem [vB95], Clarke et al. [CGP99], and Clarke and Schlingloff [CS01] for an exhaustive introduction.

Temporal logics can be classified into linear time and branching time logics [Lam80]. In linear time logics, time is considered to be of linear nature (i.e., deterministic), and in branching time logics, time is considered to be of branching nature (i.e., non-deterministic). We will now introduce the two most important representatives of linear and branching time logics in model checking, namely the *linear temporal logic LTL* and the *computation tree logic CTL*. Both logics are based on the following common *temporal operators* which are listed with their intuitive meaning:

- The *next* operator $\mathbf{X}\,\varphi$:
  Property φ holds at the next point in time.
- The *future* operator $\mathbf{F}\,\varphi$:
  Property φ holds eventually in the future.
- The *global* operator $\mathbf{G}\,\varphi$:
  Property φ holds always in the future.

- The *strong until* or simply *until* operator $\varphi \mathbf{U} \psi$:
  Property φ remains true until property ψ becomes true and property ψ must become true.
- The *weak until* or *unless* operator $\varphi \mathbf{W} \psi$:
  Property φ remains true until property ψ becomes true, but property ψ does not need to become true.
- The *release* operator $\varphi \mathbf{R} \psi$:
  Property ψ remains true until property $\varphi \wedge \psi$ becomes true, but property φ does not need to become true.

As will be shown later, some of these operators are redundant since they can be expressed by the other ones. Moreover, note that there exist several variants of them, e.g., by distinguishing between the cases if the future includes the present or not (cf. Clarke and Schlingloff [CS01]). We do not consider variants of this kind. However, in Section 2.5, we will introduce variants of the strong and weak until operator in order to increase the expressive power in the context of temporal logic *queries*.

Remark 2.1. The distinction between temporal logics using the above temporal operators and modal logics using the operators Box $\Box$ and Diamond $\Diamond$ are dealt differently with in the literature. For example, Blackburn et al. [BdRV01] consider all kinds of temporal logics as special modal logics because all of them describe properties on relational structures. On the other hand, Clarke and Schlingloff [CS01] use the term *temporal logic* only for those modal logics that contain some kind of until operator. This point of view can be justified by the fact that temporal logics with some kind of until operator are more expressive [Kam68].

2.3.1 The Linear Temporal Logic

The *linear temporal logic LTL* can be seen as an extension of propositional logic by temporal operators. Let us start with the definition of its syntax.

Definition 2.3 (LTL syntax). Let $\mathcal{A}$ be a set of atomic propositions. Then, the syntax of *LTL* is defined in Table 2.1.

$\langle LTL \rangle ::=$	$\mathcal{A}$	$\top$	$\bot$	
	$\neg \langle LTL \rangle$	$\langle LTL \rangle \wedge \langle LTL \rangle$	$\langle LTL \rangle \vee \langle LTL \rangle$	
	$\mathbf{X} \langle LTL \rangle$	$\mathbf{F} \langle LTL \rangle$	$\mathbf{G} \langle LTL \rangle$	
	$\langle LTL \rangle \mathbf{U} \langle LTL \rangle$	$\langle LTL \rangle \mathbf{W} \langle LTL \rangle$	$\langle LTL \rangle \mathbf{R} \langle LTL \rangle$	;

Table 2.1: LTL syntax

As usual in model checking, the semantics of LTL formulas is defined over paths in Kripke structures, i.e., over sequences of states. Such a definition is based on the assumptions of a starting point in time, the infinity of time, and the discrete nature of time [Eme90].[1]

Definition 2.4 (LTL semantics). The truth value of an LTL formula φ on a path π in a Kripke structure $\mathfrak{K}$, in symbols $\mathfrak{K}, \pi \models \varphi$, is defined in Table 2.2. An LTL formula φ holds in a Kripke structure $\mathfrak{K}$ with initial state s_0, in symbols $\mathfrak{K} \models \varphi$, iff $\mathfrak{K}, \pi \models \varphi$ for all $\pi \in \text{paths}(s_0)$.

For simplicity, if the Kripke structure $\mathfrak{K}$ is clear from the context, we also write $\pi \models \varphi$ instead of $\mathfrak{K}, \pi \models \varphi$. Moreover, for any LTL formula φ and set of paths Π, we write $\Pi \models \varphi$ and $\Pi \not\models \varphi$ to denote $\pi \models \varphi$ and $\pi \not\models \varphi$ respectively for all $\pi \in \Pi$. According to the semantics of the temporal operators in LTL, it is easy to verify that the following equivalences hold:

- $\mathbf{F}\varphi \Leftrightarrow \top \mathbf{U} \varphi$
- $\mathbf{G}\varphi \Leftrightarrow \varphi \mathbf{W} \bot \Leftrightarrow \bot \mathbf{R} \varphi$
- $\varphi \mathbf{W} \psi \Leftrightarrow \psi \mathbf{R} (\varphi \vee \psi)$
- $\varphi \mathbf{R} \psi \Leftrightarrow \psi \mathbf{W} (\varphi \wedge \psi)$
- $\varphi \mathbf{W} \psi \Leftrightarrow (\mathbf{G}\varphi) \vee (\varphi \mathbf{U} \psi)$
- $\varphi \mathbf{U} \psi \Leftrightarrow (\mathbf{F}\psi) \wedge (\varphi \mathbf{W} \psi)$

Remark 2.2. Note that all the above temporal operators refer to the future but not to the past. This is no restriction since LTL as well as LTL with past have the same expressive power as it was shown by Kamp [Kam68] and Gabbay [Gab89]. Hence, past operators do not increase the expressive power. However, Laroussinie et al. [LMS02] showed that temporal logics with past are exponentially more succinct.

[1] There exist also approaches for dense time, e.g., when the underlying model is isomorphic to the rational numbers. However, some equivalences between temporal logic formulas that we will use do not hold in this case (cf. Clarke and Schlingloff [CS01]).

$\mathfrak{K}, \pi \models \top$	$:\Leftrightarrow$	**true**
$\mathfrak{K}, \pi \models \bot$	$:\Leftrightarrow$	**false**
$\mathfrak{K}, \pi \models p$	$:\Leftrightarrow$	$p \in \ell(\pi(0))$
$\mathfrak{K}, \pi \models \neg\varphi$	$:\Leftrightarrow$	$\mathfrak{K}, \pi \not\models \varphi$
$\mathfrak{K}, \pi \models \varphi \wedge \psi$	$:\Leftrightarrow$	$\mathfrak{K}, \pi \models \varphi$ and $\mathfrak{K}, \pi \models \psi$
$\mathfrak{K}, \pi \models \varphi \vee \psi$	$:\Leftrightarrow$	$\mathfrak{K}, \pi \models \varphi$ or $\mathfrak{K}, \pi \models \psi$
$\mathfrak{K}, \pi \models \mathbf{X}\varphi$	$:\Leftrightarrow$	$\mathfrak{K}, \pi^1 \models \varphi$
$\mathfrak{K}, \pi \models \mathbf{F}\varphi$	$:\Leftrightarrow$	$\exists n \in \mathbb{N}.\ \mathfrak{K}, \pi^n \models \varphi$
$\mathfrak{K}, \pi \models \mathbf{G}\varphi$	$:\Leftrightarrow$	$\forall i \in \mathbb{N}.\ \mathfrak{K}, \pi^i \models \varphi$
$\mathfrak{K}, \pi \models \varphi\,\mathbf{U}\,\psi$	$:\Leftrightarrow$	$\exists n \in \mathbb{N}\ \forall i < n.\ \mathfrak{K}, \pi^i \models \varphi$ and $\mathfrak{K}, \pi^n \models \psi$
$\mathfrak{K}, \pi \models \varphi\,\mathbf{W}\,\psi$	$:\Leftrightarrow$	$\forall i \in \mathbb{N}\ \exists n \leq i.\ \mathfrak{K}, \pi^i \models \varphi$ or $\mathfrak{K}, \pi^n \models \psi$
$\mathfrak{K}, \pi \models \varphi\,\mathbf{R}\,\psi$	$:\Leftrightarrow$	$\forall i \in \mathbb{N}\ \exists n < i.\ \mathfrak{K}, \pi^i \models \psi$ or $\mathfrak{K}, \pi^n \models \varphi$

Table 2.2: LTL semantics

2.3.2 The Computation Tree Logic

In addition to linear time logics that are interpreted on computation paths, there exist also temporal logics that are interpreted on computation trees. Whereas the future in linear time logics as LTL is determined by the computation path, the future in branching time logics interpreted on computation trees is not determined, i.e., at each point in time there are several choices between possible futures. In order to be able to express such non-deterministic choices in temporal logics, two kinds of *path quantifiers* are used. They are listed in the following with their intuitive meaning:

- The *existential* path quantifier $\mathbf{E}\varphi$:
  Property φ holds in some future (i.e., φ holds possibly).
- The *universal* path quantifier $\mathbf{A}\varphi$:
  Property φ holds in all futures (i.e., φ holds necessarily).

We are now able to define the *computation tree logic CTL* introduced by Clarke and Emerson [CE82, EC82]. In CTL, every temporal operator must

$\langle CTL \rangle ::=$	$\mathcal{A}$	$\top$	$\bot$	
	$\neg \langle CTL \rangle$	$\langle CTL \rangle \wedge \langle CTL \rangle$	$\langle CTL \rangle \vee \langle CTL \rangle$	
	$\mathbf{A} \langle PF \rangle$	$\mathbf{E} \langle PF \rangle$	;	
$\langle PF \rangle ::=$	$\mathbf{X} \langle CTL \rangle$	$\mathbf{F} \langle CTL \rangle$	$\mathbf{G} \langle CTL \rangle$	
	$\langle CTL \rangle \mathbf{U} \langle CTL \rangle$	$\langle CTL \rangle \mathbf{W} \langle CTL \rangle$	$\langle CTL \rangle \mathbf{R} \langle CTL \rangle$	;

Table 2.3: CTL syntax

be immediately preceded by a path quantifier in order to select the paths on which the temporal operator has to be evaluated.

Definition 2.5 (CTL syntax). Let $\mathcal{A}$ be a set of atomic propositions. Then, the syntax of *CTL* is defined in Table 2.3.

Note that the non-terminal $\langle PF \rangle$ in Table 2.3 represents the *path formulas* in CTL. The semantics of CTL formulas is now defined over computation trees in Kripke structures, i.e., at states that are interpreted as roots of computation trees when the Kripke structure is unwound at these states.

Definition 2.6 (CTL semantics). The truth value of a CTL formula φ at state s in a Kripke structure $\mathfrak{K}$, in symbols $\mathfrak{K}, s \models \varphi$, is defined in Table 2.4. A CTL formula φ holds in a Kripke structure $\mathfrak{K}$ with initial state s_0, in symbols $\mathfrak{K} \models \varphi$, iff $\mathfrak{K}, s_0 \models \varphi$.

For simplicity, if the Kripke structure $\mathfrak{K}$ is clear from the context, we also write $s \models \varphi$ instead of $\mathfrak{K}, s \models \varphi$. Moreover, for any CTL formula φ and set of states $\mathcal{Q}$, we write $\mathcal{Q} \models \varphi$ and $\mathcal{Q} \not\models \varphi$ to denote $s \models \varphi$ and $s \not\models \varphi$ respectively for all $s \in \mathcal{Q}$. If it is clear from the context, we also write $\pi \models \varphi$ to denote $\pi(0) \models \varphi$ for any CTL formula φ and path π. Moreover, for any CTL formula φ and set of paths Π, we write $\Pi \models \varphi$ and $\Pi \not\models \varphi$ to denote $\pi(0) \models \varphi$ and $\pi(0) \not\models \varphi$ respectively for all $\pi \in \Pi$.

Remark 2.3. Although CTL is a syntactic extension of LTL by path quantifiers, both temporal logics have incomparable expressive power on Kripke structures [CD88], i.e., there are formulas in LTL that are not expressible in CTL and vice versa. However, there exist formulas that are expressible in both LTL and CTL. Maidl [Mai00] and Buccafurri et al. [BEGL01] investigated this common fragment of CTL and LTL.

$\mathfrak{K}, s \models \top$	$:\Leftrightarrow$	**true**
$\mathfrak{K}, s \models \bot$	$:\Leftrightarrow$	**false**
$\mathfrak{K}, s \models p$	$:\Leftrightarrow$	$p \in \ell(s)$
$\mathfrak{K}, s \models \neg\varphi$	$:\Leftrightarrow$	$\mathfrak{K}, s \not\models \varphi$
$\mathfrak{K}, s \models \varphi \wedge \psi$	$:\Leftrightarrow$	$\mathfrak{K}, s \models \varphi$ and $\mathfrak{K}, s \models \psi$
$\mathfrak{K}, s \models \varphi \vee \psi$	$:\Leftrightarrow$	$\mathfrak{K}, s \models \varphi$ or $\mathfrak{K}, s \models \psi$
$\mathfrak{K}, \pi \models \mathbf{X}\varphi$	$:\Leftrightarrow$	$\mathfrak{K}, \pi(1) \models \varphi$
$\mathfrak{K}, \pi \models \mathbf{F}\varphi$	$:\Leftrightarrow$	$\exists n \in \mathbb{N}.\ \mathfrak{K}, \pi(n) \models \varphi$
$\mathfrak{K}, \pi \models \mathbf{G}\varphi$	$:\Leftrightarrow$	$\forall i \in \mathbb{N}.\ \mathfrak{K}, \pi(i) \models \varphi$
$\mathfrak{K}, \pi \models \varphi\,\mathbf{U}\,\psi$	$:\Leftrightarrow$	$\exists n \in \mathbb{N}\ \forall i < n.\ \mathfrak{K}, \pi(i) \models \varphi$ and $\mathfrak{K}, \pi(n) \models \psi$
$\mathfrak{K}, \pi \models \varphi\,\mathbf{W}\,\psi$	$:\Leftrightarrow$	$\forall i \in \mathbb{N}\ \exists n \leq i.\ \mathfrak{K}, \pi(i) \models \varphi$ or $\mathfrak{K}, \pi(n) \models \psi$
$\mathfrak{K}, \pi \models \varphi\,\mathbf{R}\,\psi$	$:\Leftrightarrow$	$\forall i \in \mathbb{N}\ \exists n < i.\ \mathfrak{K}, \pi(i) \models \psi$ or $\mathfrak{K}, \pi(n) \models \varphi$
$\mathfrak{K}, s \models \mathbf{E}\varphi$	$:\Leftrightarrow$	$\exists \pi \in \mathrm{paths}(s).\ \mathfrak{K}, \pi \models \varphi$
$\mathfrak{K}, s \models \mathbf{A}\varphi$	$:\Leftrightarrow$	$\forall \pi \in \mathrm{paths}(s).\ \mathfrak{K}, \pi \models \varphi$

Table 2.4: CTL semantics

2.3.3 Other Temporal Logics

In this thesis, we are primarily concerned with the temporal logics LTL and CTL. Nevertheless, for the sake of completeness, we give a short overview of other temporal logics that are relevant in the context of model checking.

ACTL (universal CTL) [GL94] is an important sublogic of CTL obtained by allowing only universal path quantification. In order to avoid implicit existential path quantification, negations in ACTL are applied only to atomic propositions, i.e., ACTL formulas must be in negation normal form. For example, $\mathbf{AX}(\mathbf{A}(a\,\mathbf{U}\,\neg b) \wedge \mathbf{AF}\,c)$ is in ACTL. Logics that allow only universal path quantification are typically used in compositional reasoning and abstraction [CLM89, CGL94, GL94].

CTL* (extended CTL) [EH86] is a branching time logic that is more expressive than both LTL and CTL. This logic is obtained by releasing the constraint that temporal operators in CTL must be immediately preceded by a path quantifier, i.e., CTL* allows nested temporal operators without path quantifiers between them. For example, $\mathbf{EXGA}(a\,\mathbf{U}\,b)$ and $\mathbf{AFG}((a\,\mathbf{U}\,b) \vee \mathbf{EX}\,c)$ are in CTL*. Analogously to the case of CTL, the universal fragment of CTL* is called ACTL* [GL94].

CTL$^+$ [EH85] is another branching time logic that lies syntactically between CTL and CTL*. This logic is obtained by releasing the constraint on path quantification in CTL only in the case of Boolean combinations. For example, $\mathbf{AX}\,\mathbf{E}((a\,\mathbf{U}\,b) \wedge \mathbf{F}\,c)$ is in CTL$^+$. Emerson and Halpern [EH85] showed that CTL and CTL$^+$ have the same expressive power. However, it was shown by Wilke [Wil99] and Adler and Immerman [AI03] that CTL$^+$ is exponentially more succinct than CTL.

The propositional μ-calculus [EC80, Koz83] is a very expressive logic based on fixpoint definitions. Although it is not very intuitive, many temporal and program logics – including CTL* – can be encoded into the μ-calculus. In the context of symbolic model checking in Section 2.4.1, we will present the fixpoint definitions of CTL operators.

2.4 Model Checking

Model checking [CGP99, CS01] is a formal verification technique invented in the early 1980s by Clarke and Emerson [CE82, EC82] and Queille and

	Satisfiability / Validity	Model Checking
LTL	PSPACE-complete [SC85]	PSPACE-complete [SC85]
CTL	EXPTIME-complete [Eme90]	P-complete [Sch03]
CTL$^+$	2-EXPTIME-complete [JL03]	Δ_2^p-complete [LMS01]
CTL*	2-EXPTIME-complete [EJ88]	PSPACE-complete [CES86]

Table 2.5: Computational complexity

Sifakis [QS82]. Given a system model $\mathfrak{K}$ in the form of a Kripke structure and a system specification φ formulated in a temporal logic, *model checking* is the problem of algorithmically deciding $\mathfrak{K} \models \varphi$, i.e., whether the Kripke structure $\mathfrak{K}$ satisfies the temporal logic formula φ. In addition to this originally formulated model checking problem based on Kripke structures and temporal logics, model checking can also be defined for arbitrary logic formalisms: Given an interpretation $\mathfrak{M}$ of a logic formula φ, model checking is the problem of deciding whether $\mathfrak{M}$ is a model of φ.

Depending on the temporal logic in use, the computational complexity of model checking differs significantly. Table 2.5 gives an overview of the complexity of the satisfiability problem (i.e., given a formula, does there exist a model that satisfies the formula?) and the model checking problem (i.e., given both a model and a formula, does the model satisfy the formula?) for some important temporal logics. For a survey on the complexity of temporal logics see Emerson [Eme90] and Schnoebelen [Sch03].

Although these complexity results suggest that CTL model checking can be done efficiently whereas LTL model checking is practically infeasible, a refined analysis of their complexity justifies the practicability of both of them. In particular, CTL model checking can be done in time linear in both the size of the model and the length of the formula [CES86], and LTL model checking can be done in time linear in the size of the model and exponential in the length of the formula [LP85]. However, since specification formulas are rather small in practice, the exponential complexity in the length of the formula is negligible. In fact, both LTL and CTL are the most important temporal logics in formal verification by model checking.

There has been much research in the last two decades in order to find efficient model checking algorithms for LTL and CTL. For example, beside the first algorithms that are based on explicit state space labeling [EC82, CES86] and tableau construction [LP85], there exist automata-theoretic algorithms [VW86, KVW00], algorithms based on binary decision diagrams called *symbolic model checking* [BCM+92], and reductions to propositional satisfiability called *bounded model checking* [BCCZ99]. In order to increase the efficiency of model checking, there has also been much research on the combination with techniques like abstraction, compositional reasoning, partial order reductions, symmetry reductions, on-the-fly evaluation, etc. The most challenging problem in all these approaches is the *state space explosion*, i.e., the huge number of states appearing when real-world systems are modeled as Kripke structures.

In the following section, we take a closer look at symbolic model checking, since we will use symbolic algorithms in Chapter 5.

2.4.1 Symbolic Model Checking

Symbolic model checking [BCM+92, McM93] is one of the most successful approaches against the state space explosion problem. The idea is to avoid the explicit construction of the Kripke structure by encoding it into a Boolean function, which can be represented by a *binary decision diagram (BDD)* as investigated by Bryant [Bry86, Bry92]. Since BDDs are often more compact than other representations of Boolean functions, this approach allowed the verification of large systems that could not be handled until then [BCM+92, BCL+94]. Note that the practical success of symbolic model checking seems somehow surprising in consideration of the fact that symbolic LTL, CTL, and CTL* model checking is PSPACE-complete even for fixed formula length [KVW00, Sch03].

Example 2.1. Let us now demonstrate how Kripke structures can be represented by BDDs. To this aim, consider the Kripke structure shown in Figure 2.2 over the set of atomic propositions $\mathcal{A} = \{a, b\}$. It is easy to see that both states can be uniquely identified by their labelings, i.e., by the Boolean formulas $a \wedge \neg b$ and $a \wedge b$ respectively.[2] Now, in order to encode

[2] If there were states with identical labelings, we would have to add auxiliary atomic propositions until the labelings became unique.

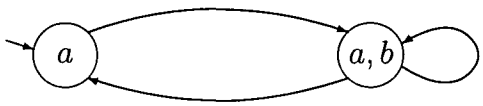


Figure 2.2: Kripke structure to be encoded as BDD

the transition relation, we have to introduce two new atomic propositions a' and b'. Each transition can then be seen as the pair consisting of its predecessor state encoded by a and b and its successor state encoded by a' and b'. In particular, an encoding of the three transitions in our example yields $a \wedge \neg b \wedge a' \wedge b'$, $a \wedge b \wedge a' \wedge \neg b'$, and $a \wedge b \wedge a' \wedge b'$. Thus, the whole Kripke structure can be encoded by

$$(a \wedge \neg b \wedge a' \wedge b') \vee (a \wedge b \wedge a' \wedge \neg b') \vee (a \wedge b \wedge a' \wedge b'). \qquad (2.1)$$

It remains to show how to represent this Boolean function by a BDD. A *binary decision diagram (BDD)* [Bry86, Bry92] is a directed acyclic graph that results from a *binary decision tree* when isomorphic subtrees are merged and unnecessary nodes are removed. A *binary decision tree* is a binary tree where each node is labeled with an atomic proposition and each edge as well as each leaf is labeled with a Boolean truth value. Boolean functions can be encoded into binary decision trees in such a way that each branch of the tree encodes an interpretation by assigning the truth value labeling an edge to the atomic proposition labeling its parent node. The resulting truth value, when applying the corresponding Boolean function to such an interpretation, is given by the labeling of the leaf.

A binary decision diagram resulting from such an encoding of our example in (2.1) is shown in Figure 2.3. It can be easily verified that the interpretations encoded into the paths that lead from the initial node to the leaf labeled with 1 are exactly the models of the formula in (2.1). All other interpretations are encoded into paths leading to the leaf labeled with 0. Note that the ordering of atomic propositions in this example is the same on each path, namely $a < a' < b < b'$. Such an ordering is the first of two properties that must be satisfied in order to obtain a canonical representation of BDDs. The second property is that all isomorphic subtrees are merged and all redundant nodes are removed. BDDs satisfying the first property are called *ordered* (OBDDs), and OBDDs satisfying the second property are

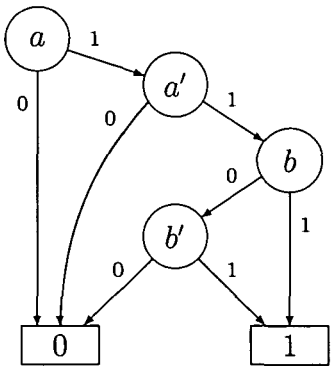


Figure 2.3: Binary decision diagram

called *reduced* (ROBDDs). The example in Figure 2.3 is therefore a *reduced ordered binary decision diagram (ROBDD)* with ordering $a < a' < b < b'$.

Obviously, ROBDDs are maximal succinct representations of BDDs for a fixed ordering of atomic propositions. The degree of succinctness, however, depends on the chosen ordering and finding the optimal ordering for maximal succinctness is coNP-complete [Bry86].

We have now shown how Kripke structures can be encoded into BDDs. In the remainder of this section, we describe how model checking can be performed based on such a representation. To this aim, we restrict ourself to CTL model checking, since symbolic model checking is primarily done for CTL and we will need this knowledge in Chapter 5.

Since BDDs encode sets of states and sets of transition relations without access to individual components, operations on BDDs have to be performed on entire sets. There exist four basic operations on BDDs:

- $\text{pre}_\exists(\mathcal{S}) = \{s \mid \exists s'.\ (s, s') \in \Delta \wedge s' \in \mathcal{S}\}$
- $\text{pre}_\forall(\mathcal{S}) = \{s \mid \forall s'.\ (s, s') \in \Delta \Rightarrow s' \in \mathcal{S}\}$
- $\text{post}_\exists(\mathcal{S}) = \{s' \mid \exists s.\ (s, s') \in \Delta \wedge s \in \mathcal{S}\}$
- $\text{post}_\forall(\mathcal{S}) = \{s' \mid \forall s.\ (s, s') \in \Delta \Rightarrow s \in \mathcal{S}\}$

Based on these operations and the fact that the powerset of the set of states of a Kripke structure forms a complete lattice under set inclusion, it is

possible to define the set of states $[\![\varphi]\!]$ satisfying the CTL formula φ by least and greatest fixpoints (in symbols, $\mu\mathcal{Z}.\tau(\mathcal{Z})$ and $\nu\mathcal{Z}.\tau(\mathcal{Z})$ respectively). The corresponding fixpoint characterization of CTL [EC80] is given by:

- $[\![p]\!] = \{s \in Q \mid p \in \ell(s)\}$
- $[\![\neg\varphi]\!] = Q \setminus [\![\varphi]\!]$
- $[\![\varphi \wedge \psi]\!] = [\![\varphi]\!] \cap [\![\psi]\!]$
- $[\![\varphi \vee \psi]\!] = [\![\varphi]\!] \cup [\![\psi]\!]$
- $[\![\mathbf{EX}\,\varphi]\!] = \mathrm{pre}_\exists([\![\varphi]\!])$
- $[\![\mathbf{AX}\,\varphi]\!] = \mathrm{pre}_\forall([\![\varphi]\!])$
- $[\![\mathbf{EG}\,\varphi]\!] = \nu\mathcal{Z}.\,([\![\varphi]\!] \cap \mathrm{pre}_\exists(\mathcal{Z}))$
- $[\![\mathbf{AG}\,\varphi]\!] = \nu\mathcal{Z}.\,([\![\varphi]\!] \cap \mathrm{pre}_\forall(\mathcal{Z}))$
- $[\![\mathbf{E}(\varphi\,\mathbf{U}\,\psi)]\!] = \mu\mathcal{Z}.\,([\![\psi]\!] \cup ([\![\varphi]\!] \cap \mathrm{pre}_\exists(\mathcal{Z})))$
- $[\![\mathbf{A}(\varphi\,\mathbf{U}\,\psi)]\!] = \mu\mathcal{Z}.\,([\![\psi]\!] \cup ([\![\varphi]\!] \cap \mathrm{pre}_\forall(\mathcal{Z})))$
- $[\![\mathbf{E}(\varphi\,\mathbf{W}\,\psi)]\!] = \nu\mathcal{Z}.\,([\![\varphi \vee \psi]\!] \cap ([\![\psi]\!] \cup \mathrm{pre}_\exists(\mathcal{Z})))$
- $[\![\mathbf{A}(\varphi\,\mathbf{W}\,\psi)]\!] = \nu\mathcal{Z}.\,([\![\varphi \vee \psi]\!] \cap ([\![\psi]\!] \cup \mathrm{pre}_\forall(\mathcal{Z})))$
- $[\![\mathbf{E}(\varphi\,\mathbf{R}\,\psi)]\!] = \nu\mathcal{Z}.\,([\![\psi]\!] \cap ([\![\varphi]\!] \cup \mathrm{pre}_\exists(\mathcal{Z})))$
- $[\![\mathbf{A}(\varphi\,\mathbf{R}\,\psi)]\!] = \nu\mathcal{Z}.\,([\![\psi]\!] \cap ([\![\varphi]\!] \cup \mathrm{pre}_\forall(\mathcal{Z})))$

It is easy to see that these fixpoint definitions are monotonic. Hence, the fixpoints always exist according to Knaster-Tarski [Tar55]. The model checking problem is then reduced to deciding if the initial state is in $[\![\varphi]\!]$.

2.5 Temporal Logic Queries

Temporal logic queries as introduced by Chan [Cha00] are a formalism motivated by database queries in order to extend model checking. The idea is to allow variables to occur in temporal logic formulas that have to be instantiated by appropriate solution formulas in such a way that the resulting formulas are satisfied by the model.

There has been active research in recent years on temporal logic queries. The starting point was the work of Chan [Cha00] who focused on a syntactic fragment of CTL queries for which he presented an efficient symbolic query solving algorithm. Afterwards, Bruns and Godefroid [BG01] generalized Chan's idea by an automata-theoretic approach that allows to solve queries of any temporal logic having a translation to alternating automata. Gurfinkel et al. [GDC02, CG03], on the other hand, investigated CTL query solving by using their multi-valued model checker χChek. At roughly the same time, Hornus and Schnoebelen [HS02] dealt with more theoretical results on the computational effort of query solving for any fragment of CTL*. Based on the author's diploma thesis [Sam02, SV03], it was then shown that Chan's original work was erroneous. We will consider all these approaches in more detail in Chapter 5.

Since temporal logic queries are a basic concept used throughout this thesis, we introduce them now more formally.

Definition 2.7 (Temporal logic query). A *temporal logic query* is a temporal logic formula where some subformulas are replaced by a special variable ?, called *placeholder*. We denote the set of LTL queries by LTLQ and the set of CTL queries by CTLQ.

It is straightforward to generalize this definition in such a way that a query can contain different variables. For simplicity, however, we restrict our considerations to queries with a single variable, namely the placeholder. In the following, we will simply write *query* instead of *temporal logic query* and *model* instead of *Kripke structure* if the definitions resp. results are independent of a particular formalism.

Definition 2.8 (Solutions). Let γ be a query, $\mathfrak{M}$ be a model, and φ be a formula. We write $\gamma[\varphi]$ to denote the result of substituting all occurrences of the placeholder in γ by φ. If $\mathfrak{M} \models \gamma[\varphi]$, then we say that φ is a *solution* to γ in $\mathfrak{M}$. We denote the set of all solutions to γ in $\mathfrak{M}$ by

$$\mathrm{sol}(\mathfrak{M}, \gamma) = \{\varphi \mid \mathfrak{M} \models \gamma[\varphi]\}.$$

Example 2.2. Consider the Kripke structure $\mathfrak{K}$ shown in Figure 2.4 and let $\gamma_1 = \mathbf{A}(a\,\mathbf{U}\,\mathbf{AX}(c \vee \mathbf{AG}\,?))$ and $\gamma_2 = \mathbf{E}(?\,\mathbf{U}\,\mathbf{AG}\,?)$ be CTL queries. It can be easily checked that $\mathfrak{K} \models \gamma_1[b \wedge \mathbf{AX}\,d]$ and $\mathfrak{K} \models \gamma_2[a \vee b]$. Hence, $b \wedge \mathbf{AX}\,d$ and $a \vee b$ are solutions to γ_1 and γ_2 respectively in $\mathfrak{K}$.

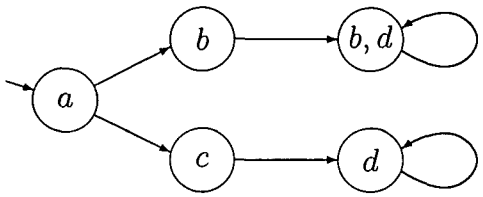


Figure 2.4: Query solving example

A query γ can be seen as a function $\gamma : \varphi \mapsto \gamma[\varphi]$ that maps formulas to formulas. This point of view leads to an important and natural property of queries, namely monotonicity. The following definition of monotonic queries originates from Chan [Cha00]. Note that *monotonic* means *monotonic increasing*; the case of *monotonic decreasing* queries is symmetric.

Definition 2.9 (Monotonic query). A query γ is *monotonic* iff $\varphi \Rightarrow \psi$ implies $\gamma[\varphi] \Rightarrow \gamma[\psi]$ for all formulas φ and ψ.

Note that this kind of monotonicity differs from that used in database theory, where a query γ is interpreted as a function $\gamma : \mathfrak{M} \mapsto \mathrm{sol}(\mathfrak{M}, \gamma)$ that maps models to the sets of its solutions in these models [AHV95]. From this point of view, a query γ is said to be *monotonic* iff

$$\mathfrak{M}_1 \subseteq \mathfrak{M}_2 \quad \text{implies} \quad \mathfrak{M}_1 \models \gamma[\varphi] \Rightarrow \mathfrak{M}_2 \models \gamma[\varphi] \tag{2.2}$$

for all models $\mathfrak{M}_1$ and $\mathfrak{M}_2$. We will now show that this kind of monotonicity is strictly stronger than monotonicity according to Definition 2.9.

Proposition 2.1. *Monotonicity according to (2.2) is strictly stronger than monotonicity according to Definition 2.9.*

Proof. First, we show that if a query γ is monotonic according to (2.2), then it is also monotonic according to Definition 2.9. To this aim, assume for the sake of contradiction that γ is not monotonic according to Definition 2.9, i.e., there exist formulas φ and ψ and a model $\mathfrak{M}$ such that $\varphi \Rightarrow \psi$ and $\mathfrak{M} \models \gamma[\varphi]$, but $\mathfrak{M} \not\models \gamma[\psi]$. W.l.o.g., we can assume that there is an atomic proposition p not occurring in $\gamma[\psi]$ such that $p \Leftrightarrow \varphi$ holds globally in $\mathfrak{M}$.[3]

[3] Otherwise, we construct a new model in such a way that we add a new atomic proposition p to $\mathfrak{M}$ such that $p \Leftrightarrow \varphi$ holds globally.

Since $\varphi \Rightarrow \psi$, we know that always when p holds in $\mathfrak{M}$ then also ψ holds in $\mathfrak{M}$ but not necessarily vice versa. Now, let $\mathfrak{M}'$ be the model resulting from $\mathfrak{M}$ by adding p such that $p \Leftrightarrow \psi$ holds globally in $\mathfrak{M}'$. Then, obviously it holds that $\mathfrak{M} \subseteq \mathfrak{M}'$. Thus, since $\mathfrak{M} \models \gamma[\varphi]$ and therefore $\mathfrak{M} \models \gamma[p]$, we obtain by (2.2) that $\mathfrak{M}' \models \gamma[p]$. This, however, is equivalent to $\mathfrak{M}' \models \gamma[\psi]$, since $p \Leftrightarrow \psi$ holds globally in $\mathfrak{M}'$. Now, recall that $\mathfrak{M}$ and $\mathfrak{M}'$ differ only in the truth values of p. Thus, since p does not occur in $\gamma[\psi]$, it follows that $\mathfrak{M} \models \gamma[\psi]$, which contradicts the assumption. Hence, monotonicity according to (2.2) implies monotonicity according to Definition 2.9.

What remains to show is that this implication does not hold in the other direction. To this aim, let $\mathfrak{M}$ be a model such that $\mathfrak{M} \not\models c$ and consider the query $\gamma = \neg c \vee ?$. It is easy to see that γ is monotonic according to Definition 2.9. For the sake of contradiction, assume that γ is also monotonic according to (2.2). Since $\mathfrak{M} \not\models c$, we know that $\mathfrak{M} \models \gamma[\bot]$. Now, let $\mathfrak{M}'$ be the model resulting from $\mathfrak{M}$ by adding c such that $\mathfrak{M}' \models c$. Then, obviously it holds that $\mathfrak{M} \subseteq \mathfrak{M}'$. Thus, since $\mathfrak{M} \models \gamma[\bot]$, we obtain by (2.2) that $\mathfrak{M}' \models \gamma[\bot]$. However, it is easy to see that $\mathfrak{M}' \not\models \gamma[\bot]$, which contradicts the assumption. Hence, γ is monotonic according to Definition 2.9 but not monotonic according to (2.2). □

Note that throughout this thesis, we will use the term *monotonic query* in the sense of Definition 2.9. Proposition 2.1 just points out a connection to database queries, but it will not be used anymore. In contrast, the following lemma will be used very frequently.

Lemma 2.1 ([Cha00]). *Let γ be a monotonic query and $\mathfrak{M}$ be a model.*

1. *The query γ has a solution in $\mathfrak{M}$ iff $\mathfrak{M} \models \gamma[\top]$.*
2. *Every formula is a solution to γ in $\mathfrak{M}$ iff $\mathfrak{M} \models \gamma[\bot]$.*

Proof. The *if* direction of *1* and the *only if* direction of *2* are trivial.
For the *only if* direction of *1* suppose that $\mathfrak{M} \models \gamma[\varphi]$ for some formula φ. Since $\varphi \Rightarrow \top$, we obtain by monotonicity $\mathfrak{M} \models \gamma[\top]$. For the *if* direction of *2* suppose that $\mathfrak{M} \models \gamma[\bot]$. Since $\bot \Rightarrow \varphi$ for every formula φ, we obtain by monotonicity $\mathfrak{M} \models \gamma[\varphi]$ for every formula φ. □

Another important and well-known property is that the composition of monotonic queries is also monotonic.

Lemma 2.2. *Let γ and γ' be monotonic queries. Then, $\gamma[\gamma']$ is monotonic.*

Proof. Let φ and ψ be formulas such that $\varphi \Rightarrow \psi$. Since γ' is monotonic, we know that $\gamma'[\varphi] \Rightarrow \gamma'[\psi]$. Thus, since γ is monotonic, we know that $\gamma[\gamma'[\varphi]] \Rightarrow \gamma[\gamma'[\psi]]$. Hence, since $\gamma[\gamma'[\theta]]$ is syntactically the same as $\gamma[\gamma'][\theta]$ for all formulas θ, we obtain $\gamma[\gamma'][\varphi] \Rightarrow \gamma[\gamma'][\psi]$, i.e., $\gamma[\gamma']$ is monotonic. □

2.5.1 Additional Temporal Operators

In Chapter 4, we will investigate syntactic fragments of temporal logic queries where only a single occurrence of the placeholder is allowed. For example, query γ_1 in Example 2.2 is such a query whereas γ_2 is not. In order not to loose all queries with multiple occurrences of the placeholder, we introduce four additional temporal operators following Chan [Cha00]:

- The *overlapping strong until* operator: $\varphi \,\mathring{\mathbf{U}}\, \psi = \varphi \,\mathbf{U}\, (\varphi \wedge \psi)$
- The *disjoint strong until* operator: $\varphi \,\bar{\mathbf{U}}\, \psi = \varphi \,\mathbf{U}\, (\neg\varphi \wedge \psi)$
- The *overlapping weak until* operator: $\varphi \,\mathring{\mathbf{W}}\, \psi = \varphi \,\mathbf{W}\, (\varphi \wedge \psi)$
- The *disjoint weak until* operator: $\varphi \,\bar{\mathbf{W}}\, \psi = \varphi \,\mathbf{W}\, (\neg\varphi \wedge \psi)$

Of course, these temporal operators do not increase the expressive power of LTL, CTL, and CTL*. However, they enable us to express a certain class of temporal logic queries with multiple occurrences of the placeholder. For example, the LTL query $? \,\mathbf{U}\, (? \wedge \psi)$ would not be expressible by using the standard temporal operators when only a single occurrence of the placeholder is allowed. By using the overlapping strong until operator, however, we are able to express the query $? \,\mathring{\mathbf{U}}\, \psi = ? \,\mathbf{U}\, (? \wedge \psi)$.

Based on the equivalences in Section 2.3.1, it is easy to verify that also the following equivalences hold:

- $\varphi \,\mathring{\mathbf{W}}\, \psi \Leftrightarrow (\mathbf{G}\,\varphi) \vee (\varphi \,\mathring{\mathbf{U}}\, \psi)$
- $\varphi \,\bar{\mathbf{W}}\, \psi \Leftrightarrow (\mathbf{G}\,\varphi) \vee (\varphi \,\bar{\mathbf{U}}\, \psi)$
- $\varphi \,\mathbf{W}\, \psi \Leftrightarrow (\varphi \vee \psi) \,\mathring{\mathbf{W}}\, \psi$
- $\varphi \,\mathring{\mathbf{U}}\, \psi \Leftrightarrow (\mathbf{F}\,\psi) \wedge (\varphi \,\mathring{\mathbf{W}}\, \psi)$
- $\varphi \,\bar{\mathbf{U}}\, \psi \Leftrightarrow (\mathbf{F}\,\psi) \wedge (\varphi \,\bar{\mathbf{W}}\, \psi)$
- $\varphi \,\mathbf{U}\, \psi \Leftrightarrow (\varphi \vee \psi) \,\mathring{\mathbf{U}}\, \psi$

Note that $\varphi \,\mathring{\mathbf{W}}\, \psi = \varphi \,\mathbf{W}\, (\varphi \wedge \psi) \Leftrightarrow \psi \,\mathbf{R}\, \varphi$, i.e., the overlapping weak until operator is the same as the release operator with operands swapped.

Hence, when using the additional temporal operators above, the release operator can be omitted. Moreover, note that the additional temporal operators cover some fragment of CTL$^+$ when used in CTL. For example, $\mathbf{A}(\varphi \,\mathring{\mathbf{W}}\, \psi) \Leftrightarrow \mathbf{A}((\mathbf{G}\,\varphi) \vee (\varphi \,\mathring{\mathbf{U}}\, \psi))$ and $\mathbf{E}(\varphi \,\bar{\mathbf{U}}\, \psi) \Leftrightarrow \mathbf{E}((\mathbf{F}\,\psi) \wedge (\varphi \,\bar{\mathbf{W}}\, \psi))$.

Negation. In addition to a restriction to a single occurrence of the placeholder, we will also consider queries in *negation normal form (NNF)* in Chapter 4, i.e., queries where negation appears only in front of atomic propositions and the placeholder. However, in order to be able to transform any temporal logic query into NNF and therefore to preserve expressive power, the chosen query language must be closed under negation, i.e., the negation of each operator in the language can be expressed by other operators in the language. The following list shows the equivalences between temporal operators and their dual operators concerning negation:

- $\neg\mathbf{F}\,\varphi \Leftrightarrow \mathbf{G}\,\neg\varphi$
- $\neg\mathbf{G}\,\varphi \Leftrightarrow \mathbf{F}\,\neg\varphi$
- $\neg(\varphi\,\mathbf{U}\,\psi) \Leftrightarrow \neg\varphi\,\mathbf{R}\,\neg\psi$
- $\neg(\varphi\,\mathbf{R}\,\psi) \Leftrightarrow \neg\varphi\,\mathbf{U}\,\neg\psi$
- $\neg(\varphi\,\mathbf{U}\,\psi) \Leftrightarrow \neg\psi\,\mathring{\mathbf{W}}\,\neg\varphi$
- $\neg(\varphi\,\mathring{\mathbf{W}}\,\psi) \Leftrightarrow \neg\psi\,\mathbf{U}\,\neg\varphi$
- $\neg(\varphi\,\mathbf{W}\,\psi) \Leftrightarrow \neg\psi\,\mathring{\mathbf{U}}\,\neg\varphi$
- $\neg(\varphi\,\mathring{\mathbf{U}}\,\psi) \Leftrightarrow \neg\psi\,\mathbf{W}\,\neg\varphi$

These equivalences follow directly from the definitions of the operators and can be easily verified. An additional equivalence not shown above is that of the next operator, which is dual to itself, that is, $\neg\mathbf{X}\,\varphi \Leftrightarrow \mathbf{X}\,\neg\varphi$.

Special cases are the disjoint strong and weak until operators, which do not have a dual operator in the above sense in our language. However, their negations can be expressed by:

- $\neg(\varphi\,\bar{\mathbf{U}}\,\psi) \Leftrightarrow (\varphi \vee \neg\psi)\,\mathring{\mathbf{W}}\,\neg\varphi$
- $\neg(\varphi\,\bar{\mathbf{W}}\,\psi) \Leftrightarrow (\varphi \vee \neg\psi)\,\mathring{\mathbf{U}}\,\neg\varphi$

Note that the first argument of both operators is duplicated after negation. In particular, this means that if we allow only a single occurrence of the placeholder, then it must not occur in the first argument of these operators if they are under the scope of negation. Otherwise, there would be several occurrences of the placeholder after building the NNF, e.g., $\neg\mathbf{E}(?\,\bar{\mathbf{U}}\,\psi) \Leftrightarrow \mathbf{A}((? \vee \neg\psi)\,\mathring{\mathbf{W}}\,\neg?)$. Of course, we could overcome this restriction by introducing new operators; however, placeholders in the first

arguments of the disjoint strong and weak until operators will also be forbidden for monotonicity reasons.

Monotonicity. Now, let us consider a further property of these operators, namely monotonicity. The easiest way to obtain monotonic queries is to construct them by composing queries that are already known to be monotonic (cf. Lemma 2.2). Thus, since the simplest queries to be composed are those obtained from the temporal operators when replacing some arguments by the placeholder, it is necessary to know which of them are monotonic.

This, however, can be easily proved. For example, to show that the strong until operator is monotonic in its second argument, consider the query $\gamma = \theta \,\mathbf{U}\, ?$ and assume that $\varphi \Rightarrow \psi$ as well as $\pi \models \gamma[\varphi]$ for any formulas φ and ψ and any path π. By the definition of the until operator, this means that there exists $n \in \mathbb{N}$ such that $\pi^n \models \varphi$ and for all $i < n$ it holds that $\pi^i \models \theta$. Thus, since $\varphi \Rightarrow \psi$, we know that $\pi^n \models \psi$ and therefore $\pi \models \gamma[\psi]$. Hence, the strong until operator is monotonic in its second argument.

In the same way it can be easily shown that almost all temporal operators introduced in this chapter are monotonic in all their arguments. The only exceptions are the disjoint strong and the disjoint weak until operator, which are not monotonic in their first argument as shown in the following example.

Example 2.3. Let $\gamma_1 = ? \,\bar{\mathbf{U}}\, c$ and $\gamma_2 = ? \,\bar{\mathbf{W}}\, c$. Moreover, let π be a path such that $\ell(\pi(0)) = \{a, c\}$ and $\ell(\pi(i)) = \emptyset$ for all $i \geq 1$. Then, it can be easily verified that $\pi \models \gamma_1[a \wedge b]$ as well as $\pi \models \gamma_2[a \wedge b]$, but $\pi \not\models \gamma_1[a]$ and $\pi \not\models \gamma_2[a]$ although $a \wedge b \Rightarrow a$. Note that this is a counterexample to monotonicity in the sense of *monotonically increasing*. In order to show that both queries are not monotonically decreasing either, let π be a path such that $\ell(\pi(0)) = \{a\}$ and $\ell(\pi(i)) = \{c\}$ for all $i \geq 1$. Then, it can be easily verified that $\pi \models \gamma_1[a \vee b]$ as well as $\pi \models \gamma_2[a \vee b]$, but $\pi \not\models \gamma_1[b]$ and $\pi \not\models \gamma_2[b]$ although $b \Rightarrow a \vee b$. Hence, γ_1 and γ_2 are neither monotonically increasing nor monotonically decreasing.

Validity. Finally, we consider a property that we will need in Chapter 5 in the context of query solving algorithms. Chan [Cha00] presented a symbolic algorithm for solving queries in his syntactic fragment of *valid* CTL queries. A necessary condition for validity according to Chan is the existence of a

solution in every Kripke structure, i.e., the validity of $\gamma[\top]$ is a necessary condition for the validity of γ (cf. Lemma 2.1). We will now show how to restrict CTL queries syntactically in order to guarantee the existence of a solution in every Kripke structure. In fact, queries composed of the following "elementary" temporal logic queries always have a solution as can be easily proved by structural induction [Sam02]:

- $\varphi \vee ?$
- $\mathbf{X}?$
- $\mathbf{G}?$
- $\varphi \mathbf{U}?$
- $?\,\mathbf{W}\,\varphi$
- $\varphi \mathbf{W}?$
- $?\,\mathring{\mathbf{W}}\,\varphi$
- $\varphi \bar{\mathbf{W}}?$

For example, the queries $\mathbf{X}((\varphi\,\mathbf{U}\,?)\,\mathbf{W}\,\psi)$ and $\varphi\,\mathbf{U}\,\mathbf{G}(?\,\mathring{\mathbf{W}}\,\psi)$ always have a solution. Therefore, in order to obtain a subset of a given set of queries such that each query in this subset always has a solution, it suffices to restrict the given set to those queries that are composed of the above operators. Note that this is only a sufficient condition, i.e., there may also exist other queries that always have a solution.

2.6 Summary

The basic formalism for modelling systems in model checking are *Kripke structures*, i.e., transition graphs with labeled nodes. Each node represents a system state at a certain point in time and the atomic propositions labeling the nodes represent the system properties at these states. Kripke structures model the system behavior over time and their properties can be expressed in logic formalisms called *temporal logics*. The most important temporal logics in model checking and also in the context of this thesis are the *linear temporal logic LTL* and the *computation tree logic CTL*. Given a Kripke structure and a temporal logic formula, *model checking* is to decide whether the Kripke structure satisfies the formula.

There exist several approaches and algorithms for answering this question. *Symbolic model checking* is one of the most successful ones against state space explosion problem. The idea behind this approach is to encode Kripke structures into binary decision diagrams (BDDs) and to reduce the model checking problem to the computation of fixpoints over sets of states.

An extension of model checking in the sense that formulas are not only checked against the model but are extracted from the model under certain

constraints is known by the term *temporal logic queries*, that are temporal logic formulas containing variables that have to be instantiated in such a way that the model satisfies the resulting formula. Temporal logic queries are a fundamental concept used throughout this thesis.

Chapter 3

Circular Compositions

3.1 Introduction

Compositional reasoning is a well-known technique against the state space explosion problem. Intuitively, compositional reasoning means to decompose a large system into small subsystems that can be handled at once. After verifying (or extracting) properties of these subsystems, they are composed according to the compositional reasoning rules in use. In this way, specifications of a large system can be deduced from the specifications of its subsystems without constructing the large system explicitly.

A simple example of such a compositional reasoning rule is the sequential composition rule in Hoare logic. A much more challenging task, however, is the composition of parallel resp. concurrent systems, since such systems may depend on each other in a circular manner. Hence, composition rules for concurrent systems must be able to handle some kind of circularity. This kind of reasoning is therefore called *circular compositional reasoning.*

Since, in general, properties of systems that depend on other systems cannot be proved completely independently, it is necessary to make assumptions on their respective environments (i.e., on the behavior of systems on which they depend). Therefore, the expression $\langle\varphi\rangle\,\mathfrak{X}\,\langle\psi\rangle$ in compositional reasoning denotes that each system containing $\mathfrak{X}$ as subsystem guarantees property ψ under the assumption φ. Let

$$\langle\varphi_2 \wedge \varphi_3\rangle\,\mathfrak{X}_1\,\langle\psi_1\rangle \qquad \langle\varphi_1 \wedge \varphi_3\rangle\,\mathfrak{X}_2\,\langle\psi_2\rangle \qquad \langle\varphi_2\rangle\,\mathfrak{X}_3\,\langle\psi_3\rangle$$

be such expressions denoting that the three systems $\mathfrak{X}_1$, $\mathfrak{X}_2$, and $\mathfrak{X}_3$ guarantee properties ψ_1, ψ_2, and ψ_3 respectively under the corresponding assumptions. For example, $\mathfrak{X}_2$ guarantees property ψ_2 under assumption φ_1

on $\mathfrak{X}_1$ and assumption φ_3 on $\mathfrak{X}_3$. Consequently, system $\mathfrak{X}_1$ depends on $\mathfrak{X}_2$ and $\mathfrak{X}_3$, system $\mathfrak{X}_2$ depends on $\mathfrak{X}_1$ and $\mathfrak{X}_3$, and system $\mathfrak{X}_3$ depends on $\mathfrak{X}_2$.

The task of circular compositional reasoning is now to determine which properties the composed system $\mathfrak{X}_1 \| \mathfrak{X}_2 \| \mathfrak{X}_3$ satisfies. In the ideal case, it can be concluded that the formula

$$\langle \top \rangle \, \mathfrak{X}_1 \| \mathfrak{X}_2 \| \mathfrak{X}_3 \, \langle \psi_1 \wedge \psi_2 \wedge \psi_3 \rangle$$

holds, i.e., the composed system satisfies ψ_1, ψ_2, and ψ_3 without assumptions. Because of the circularity of the dependencies, however, such compositions are unsound in general. Therefore, in order to guarantee the soundness of circular compositions, the dependencies between and the assumptions on the systems have to satisfy certain conditions. There exist several approaches in the literature to identify such conditions.

In this chapter, we consider circular composition rules from an abstract point of view. In particular, we present a generic rule for a certain form of assumptions and we prove its soundness. This composition rule can be successively applied in order to compose circular dependent subsystems and to remove assumptions in a systematic way. We believe that our abstract considerations provide us with a new point of view on circular compositions.

Our original motivation for the work presented in this chapter, however, comes from proof-theoretic applications. In particular, the system identifiers $\mathfrak{X}_1$, $\mathfrak{X}_2$, and $\mathfrak{X}_3$ above can also be seen as proof identifiers. Then, the above expressions denote that property ψ_1 was proved under the assumptions φ_2 and φ_3, property ψ_2 was proved under the assumptions φ_1 and φ_3, and property ψ_3 was proved under the assumption φ_2. Consequently, compositional reasoning in this context means to compose auxiliary results obtained by subproofs to a main result without proving it directly.

It is easy to see that the sequential composition rule under this interpretation corresponds to the classical *cut rule* in logic calculi. Motivated by our proof-theoretic application of composing circular dependent lemmas, we therefore investigate a *circular cut rule* in logic calculi corresponding to a circular compositional reasoning rule. The basic idea behind this rule will be used in Chapter 4 in order to compose auxiliary results that circularly depend on each other. In particular, our auxiliary results will be composed to stronger results while the assumptions are successively removed until we obtain our main result without assumptions.

This chapter is organized as follows: In Section 3.2, we investigate our circular cut rule as circular counterpart to the classical cut rule in logic calculi. Afterwards, in Section 3.3, we show how the circular cut rule can be applied in mutual inductive proofs. In particular, we present the non-compositional proof method in Section 3.3.1 and the compositional proof method using the circular cut rule in Section 3.3.2. Compositional reasoning in general is then covered by Section 3.4. Starting with an abstract non-circular composition rule, we show that the circular case is unsound in general and that a sound circular composition rule can be obtained in analogy to the circular cut rule. In Section 3.5, we give a short overview of related work. Finally, we summarize in Section 3.6.

3.2 A Circular Cut Rule

In this section, we present a compositional proof rule called *circular cut rule*. This name arises from two reasons: First, written as proof rule in a logic calculus, it reminds one of the classical cut rule:

$$\frac{\Gamma \vdash \alpha, \Delta \quad \Sigma, \alpha \vdash \Pi}{\Gamma, \Sigma \vdash \Delta, \Pi}\ cut$$

In particular, it cuts out assumptions when composing properties that depend on each other in a certain circular form. This leads to the second reason: It enables us to resolve some kind of circular arguments. In fact, the formal expressions handled by the proof rule are only circular at the first sight (not surprisingly, since circular arguments are unsound in general). Actually, the circularities we consider are of a *spiral* kind as illustrated in Figure 3.1. We will show that circular arguments based on such spiral dependencies are sound. This result is easy to verify and was already mentioned by several authors in the context of concurrency verification as will be summarized in Section 3.5. In the following, we will consider this property from a proof-theoretic point of view, since such kinds of circular dependencies appear very frequently in our proofs in Chapter 4.

In order to be able to prove the soundness of circular compositions by using our circular cut rule, we need the following property.

Definition 3.1 (Downward closed). Let n be a variable over $\mathbb{N}$. A formula φ is *downward closed* in n iff $\varphi[n/c+1] \Rightarrow \varphi[n/c]$ for all $c \in \mathbb{N}$.

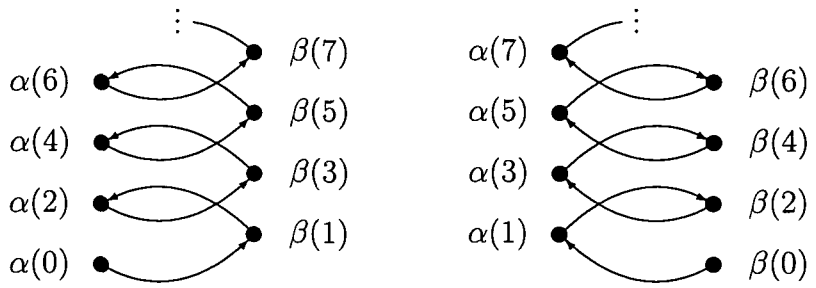


Figure 3.1: Spiral dependencies

Now, the following theorem presents our circular cut rule and states its soundness. We use a sequent calculus style proof for better readability.

Theorem 3.1. *The circular cut rule*

$$\frac{\Gamma, \forall i < n_1.\, \beta(i) \vdash \alpha(n_1), \Delta \quad \Sigma, \forall i < n_2.\, \alpha(i) \vdash \beta(n_2), \Pi}{\Gamma[n_1/n], \Sigma[n_2/n] \vdash \alpha(n) \wedge \beta(n), \Delta, \Pi}\ ccut$$

is sound if all formulas in Γ *are downward closed in* n_1*, all formulas in* Σ *are downward closed in* n_2*, no formula in* Δ *contains* n_1*, no formula in* Π *contains* n_2*, and no formula in* $\Gamma \cup \Sigma \cup \Delta \cup \Pi \cup \{\alpha, \beta\}$ *contains* n*.*

Proof. Induction on $n \in \mathbb{N}$.

Induction start: Since $\forall i < 0.\, \alpha(i)$ and $\forall i < 0.\, \beta(i)$ are trivially true, we obtain the following proof of the circular cut rule for the case $n = 0$.

$$\dfrac{\dfrac{\dfrac{\Gamma, \forall i < n_1.\, \beta(i) \vdash \alpha(n_1), \Delta}{\Gamma[n_1/0], \forall i < 0.\, \beta(i) \vdash \alpha(0), \Delta} \quad \dfrac{\Sigma, \forall i < n_2.\, \alpha(i) \vdash \beta(n_2), \Pi}{\Sigma[n_2/0], \forall i < 0.\, \alpha(i) \vdash \beta(0), \Pi}}{\Gamma[n_1/0] \vdash \alpha(0), \Delta \quad \Sigma[n_2/0] \vdash \beta(0), \Pi}}{\Gamma[n_1/0], \Sigma[n_2/0] \vdash \alpha(0) \wedge \beta(0), \Delta, \Pi}\ \wedge r$$

Induction step: Suppose that the inference rule is sound for all $n \leq c$ and we have inferred $\Gamma[n_1/n], \Sigma[n_2/n] \vdash \alpha(n) \wedge \beta(n), \Delta, \Pi$. Thus, by successively applying weakening right, conjunction right, and contraction left, we obtain $\Gamma[n_1/c], \Sigma[n_2/c] \vdash \forall i \leq c.\, \beta(i), \Delta, \Pi$ and $\Gamma[n_1/c], \Sigma[n_2/c] \vdash \forall i \leq c.\, \alpha(i), \Delta, \Pi$. The entire inference step from $n \leq c$ to $n = c + 1$ is then shown in Figure 3.2. □

$$\dfrac{\dfrac{\textbf{IH: for all } n \leq c : \Gamma[n_1/n], \Sigma[n_2/n] \vdash \alpha(n) \wedge \beta(n), \Delta, \Pi}{\vdots}}{\dfrac{\Gamma[n_1/c], \Sigma[n_2/c] \vdash \forall i \leq c.\, \beta(i), \Delta, \Pi \qquad \dfrac{\Gamma, \forall i < n_1.\, \beta(i) \vdash \alpha(n_1), \Delta}{\Gamma[n_1/c+1], \forall i \leq c.\, \beta(i) \vdash \alpha(c+1), \Delta}\, cut}{\dfrac{\Gamma[n_1/c], \Gamma[n_1/c+1], \Sigma[n_2/c] \vdash \alpha(c+1), \Delta, \Pi}{\Gamma[n_1/c+1], \Sigma[n_2/c] \vdash \alpha(c+1), \Delta, \Pi}\, cl}\, cl}$$
$$(1)$$

$$\dfrac{\dfrac{\textbf{IH: for all } n \leq c : \Gamma[n_1/n], \Sigma[n_2/n] \vdash \alpha(n) \wedge \beta(n), \Delta, \Pi}{\vdots}}{\dfrac{\Gamma[n_1/c], \Sigma[n_2/c] \vdash \forall i \leq c.\, \alpha(i), \Delta, \Pi \qquad \dfrac{\Sigma, \forall i < n_2.\, \alpha(i) \vdash \beta(n_2), \Pi}{\Sigma[n_2/c+1], \forall i \leq c.\, \alpha(i) \vdash \beta(c+1), \Pi}\, cut}{\dfrac{\Gamma[n_1/c], \Sigma[n_2/c], \Sigma[n_2/c+1] \vdash \beta(c+1), \Delta, \Pi}{\Gamma[n_1/c], \Sigma[n_2/c+1] \vdash \beta(c+1), \Delta, \Pi}\, cl}\, cl}$$
$$(2)$$

$$\dfrac{\dfrac{(1) \qquad (2)}{\Gamma[n_1/c], \Gamma[n_1/c+1], \Sigma[n_2/c], \Sigma[n_2/c+1] \vdash \alpha(c+1) \wedge \beta(c+1), \Delta, \Pi}\, \wedge r}{\Gamma[n_1/c+1], \Sigma[n_2/c+1] \vdash \alpha(c+1) \wedge \beta(c+1), \Delta, \Pi}\, cl$$

Figure 3.2: Induction step of the proof of Theorem 3.1

Of course, the circular cut rule can be generalized to any well-founded set. For our purposes, however, it is sufficient to consider the special case of natural numbers. We will now show how the circular cut rule can be applied in mutual inductive proofs.

3.3 Mutual Induction

Mutual induction is a variant of mathematical induction for proving several statements that depend on each other in a circular manner. Since we will use mutual induction in combination with other inductive proof methods, let us start with a short overview.

Mathematical induction [GT96, HMU01] is a very important method for proving mathematical statements of the form $\mathcal{S}(n)$, where $n \in \mathbb{N}$. In its most common form, it consists of proving:

1. $\mathcal{S}(n)$ holds true for $n = 0$.
2. $\forall n \in \mathbb{N}$: If $\mathcal{S}(n)$ holds true, then $\mathcal{S}(n+1)$ holds true.

Then, the induction principle based on the fifth Peano axiom allows us to conclude that $\mathcal{S}(n)$ holds for all $n \in \mathbb{N}$.

There exist several equivalent variants of this inductive proof method. For example, proving a statement $\mathcal{S}(n)$ for all $n \geq n_0$, can be simply done by applying the above induction method to the statement $\mathcal{S}'(n) = \mathcal{S}(n_0 + n)$ for all $n \in \mathbb{N}$. It is therefore easy to see that the above conditions for induction start and induction step can be reformulated in order to prove statements of the form $\mathcal{S}(n)$, where $n \geq n_0$.

Another well known and equivalent inductive proof method is called *strong induction*. For proving a statement $\mathcal{S}(n)$ for all $n \in \mathbb{N}$, it requires:

$$\forall n \in \mathbb{N}\text{: If } \mathcal{S}(i) \text{ holds true for all } i < n, \text{ then } \mathcal{S}(n) \text{ holds true.}$$

Note that this property combines induction start and induction step in a single condition. In order to see this, consider the case $n = 0$. Then, the antecedent of the condition is trivially satisfied and therefore $\mathcal{S}(n)$ must be true for $n = 0$, which yields the induction start above. Note that we have already used the principle of strong induction in the formulation of the circular cut rule in Section 3.2.

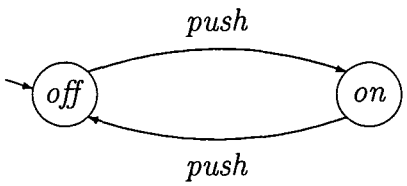


Figure 3.3: On/off switch

A generalization of the inductive proof method based on a well-founded domain instead of natural numbers is called *structural induction*. Although structural inductive proofs can be reduced to proofs by induction on natural numbers, it is often more compact and readable to use the original domain. Proofs by structural induction can be applied to all kinds of domains given by inductive definitions, e.g., data structures, formal languages, recursive functions, etc. In Chapter 4, we will frequently use structural induction to prove properties of query languages defined by context-free grammars.

All the mentioned variants of mathematical induction can be used in combination with a proof method called *mutual induction*. Although mutual induction is not more powerful than the methods described above, it enables us to prove statements of the form

$$\mathcal{S}(n) = \mathcal{S}_1(n) \wedge \mathcal{S}_2(n) \wedge \ldots \wedge \mathcal{S}_k(n)$$

in a more structured way. In particular, we consider $\mathcal{S}(n)$ as a collection of statements $\mathcal{S}_1(n), \mathcal{S}_2(n), \ldots, \mathcal{S}_k(n)$ which we want to prove simultaneously and which mutually depend on each other. We illustrate the classical approach of how to prove this by a simple example presented in [HMU01].

3.3.1 Non-compositional

Consider the automaton of an on/off switch shown in Figure 3.3. Initially, the automaton is in state *off*. Then, by each *push*-action, it switches between *on* and *off*. Therefore, the functionality of the automaton can be characterized by the statement $\mathcal{S}(n) = \mathcal{S}_1(n) \wedge \mathcal{S}_2(n)$, where

- $\mathcal{S}_1(n)$: The automaton is in state *off* after n *push*-actions iff n is even.
- $\mathcal{S}_2(n)$: The automaton is in state *on* after n *push*-actions iff n is odd.

Obviously, $\mathcal{S}_1(n)$ is a statement about state *off* and $\mathcal{S}_2(n)$ is a statement about state *on*. Since both states occur within a cycle in the automaton, the corresponding statements depend on each other in the sense that if $\mathcal{S}_1(n)$ is true for a fixed n, then $\mathcal{S}_2(n+1)$ is true, and if $\mathcal{S}_2(n)$ is true for a fixed n, then $\mathcal{S}_1(n+1)$ is true. Because of these dependencies, we show the truth of statement $\mathcal{S}(n)$ by a single inductive proof. However, in order to make the proof more structured, we consider $\mathcal{S}(n)$ as the collection of $\mathcal{S}_1(n)$ and $\mathcal{S}_2(n)$ and divide the induction start and the induction step accordingly. This is now exemplified in the proof of the following theorem [HMU01].

Theorem 3.2 (Example). *The statement $\mathcal{S}(n)$ holds true for all $n \in \mathbb{N}$.*

Proof. Induction on $n \in \mathbb{N}$.

Induction start:

- $\mathcal{S}_1(0)$: The *if* direction is trivially satisfied, since the automaton is in state *off* after 0 *push*-actions. The *only if* direction is also trivially satisfied, since 0 is an even number.
- $\mathcal{S}_2(0)$: The *if* direction is trivially satisfied, since 0 is not an odd number. The *only if* direction is also trivially satisfied, since the automaton cannot be in state *on* after 0 *push*-actions.

Induction step:

- $\mathcal{S}_1(n+1)$: For the *if* direction, assume that $n+1$ is an even number. Hence, n is an odd number and we obtain by induction hypothesis that the automaton is in state *on* after n *push*-actions. Since the automaton switches by a single *push*-action from state *on* to state *off*, we know that it is in state *off* after $n+1$ *push*-actions.
  For the *only if* direction, assume that the automaton is in state *off* after $n+1$ *push*-actions. Since there is only one possibility to switch to state *off* by a single *push*-action, we know that the automaton is in state *on* after n *push*-actions. Hence, we obtain by induction hypothesis that n is an odd number, which trivially implies that $n+1$ is an even number.
- $\mathcal{S}_2(n+1)$: For the *if* direction, assume that $n+1$ is an odd number. Hence, n is an even number and we obtain by induction hypothesis that

the automaton is in state *off* after n *push*-actions. Since the automaton switches by a single *push*-action from state *off* to state *on*, we know that it is in state *on* after $n + 1$ *push*-actions.
For the *only if* direction, assume that the automaton is in state *on* after $n+1$ *push*-actions. Since there is only one possibility to switch to state *on* by a single *push*-action, we know that the automaton is in state *off* after n *push*-actions. Hence, we obtain by induction hypothesis that n is an even number, which trivially implies that $n + 1$ is an odd number.

This concludes the proof. □

In addition to splitting the induction start and the induction step according to the substatements of $\mathcal{S}(n)$, we can also split the whole proof into subproofs and compose them afterwards.

3.3.2 Compositional

A mutual inductive proof in the form as presented in Section 3.3.1 may be easily comprehensible for this simple example. For more complex statements consisting of many substatements with complex dependencies, however, a decomposition into subproofs would be desirable. To this aim, the truth of each substatement has to be proved under the assumption of the truth of the other substatements. Afterwards, the obtained results are composed by further proofs in order to remove the auxiliary assumptions.

Let as illustrate this by the same example as in Section 3.3.1, where $\mathcal{S}(n) = \mathcal{S}_1(n) \wedge \mathcal{S}_2(n)$. We start by proving statement $\mathcal{S}_1(n)$ under the assumption of the truth of statement $\mathcal{S}_2(n)$.

Lemma 3.1 (Example). *Let $n \in \mathbb{N}$ and suppose that $\mathcal{S}_2(i)$ holds true for all natural numbers $i < n$. Then, $\mathcal{S}_1(n)$ holds true.*

Proof. For the *if* direction, assume that n is an even number. If $n = 0$, the automaton must be in the initial state *off*. Otherwise, if $n > 0$, we know that $n - 1$ is an odd number. Hence, by assumption on $\mathcal{S}_2$, it follows that the automaton is in state *on* after $n - 1$ *push*-actions. Since the automaton switches by a single *push*-action from state *on* to state *off*, it must be in state *off* after n *push*-actions.

For the *only if* direction, assume that the automaton is in state *off* after n *push*-actions. If $n = 0$, we trivially know that n is even. Otherwise, if $n > 0$,

the automaton must be in state *on* after $n-1$ *push*-actions, since there is only one possibility to switch to state *off* by a single *push*-action. Hence, by assumption on $\mathcal{S}_2$, we know that $n-1$ is an odd number, which trivially implies that n is an even number. □

Analogously, in the following lemma we prove statement $\mathcal{S}_2(n)$ under the assumption of the truth of statement $\mathcal{S}_1(n)$.

Lemma 3.2 (Example). *Let $n \in \mathbb{N}$ and suppose that $\mathcal{S}_1(i)$ holds true for all natural numbers $i < n$. Then, $\mathcal{S}_2(n)$ holds true.*

Proof. For the *if* direction, assume that n is an odd number. Thus, we know that $n-1$ is an even number. Hence, by assumption on $\mathcal{S}_1$, it follows that the automaton is in state *off* after $n-1$ *push*-actions. Since the automaton switches by a single *push*-action from state *off* to state *on*, it must be in state *on* after n *push*-actions.

For the *only if* direction, assume that the automaton is in state *on* after n *push*-actions. Since the initial state is *off*, we know that $n > 0$. Thus, the automaton must be in state *off* after $n-1$ *push*-actions, since there is only one possibility to switch to state *on* by a single *push*-action. Hence, by assumption on $\mathcal{S}_1$, we know that $n-1$ is an even number, which trivially implies that n is an odd number. □

Now, we obtain the truth of $\mathcal{S}(n) = \mathcal{S}_1(n) \wedge \mathcal{S}_2(n)$ for all $n \in \mathbb{N}$ by the composition of Lemma 3.1 and Lemma 3.2. There are two possibilities to do this: The first one is to use an inductive proof on $n \in \mathbb{N}$ directly, and the second one is to use the circular cut rule from Section 3.2. Both variants are presented in the following.

Theorem 3.2 (Example). *The statement $\mathcal{S}(n)$ holds true for all $n \in \mathbb{N}$.*

Proof. Induction on $n \in \mathbb{N}$.
Induction start: If $n = 0$, then the assumption of Lemma 3.1 on $\mathcal{S}_2$ and the assumption of Lemma 3.2 on $\mathcal{S}_1$ are trivially satisfied. Hence, by Lemma 3.1 and Lemma 3.2, we know that $\mathcal{S}_1(0)$ and $\mathcal{S}_2(0)$ are true. Thus, their conjunction $\mathcal{S}(0) = \mathcal{S}_1(0) \wedge \mathcal{S}_2(0)$ holds.
Induction step: Let $n \in \mathbb{N}$ and assume that $\mathcal{S}(i)$ holds for all natural numbers $i \leq n$. Then, both $\mathcal{S}_1(i)$ and $\mathcal{S}_2(i)$ hold for all $i < n+1$. Hence, by Lemma 3.1 and Lemma 3.2, we know that both $\mathcal{S}_1(n+1)$ and $\mathcal{S}_2(n+1)$ are true. Thus, their conjunction $\mathcal{S}(n+1) = \mathcal{S}_1(n+1) \wedge \mathcal{S}_2(n+1)$ holds. □

Proof. Application of the circular cut rule.

$$\frac{\forall i < n_1.\, \mathcal{S}_2(i) \vdash \mathcal{S}_1(n_1) \quad \forall i < n_2.\, \mathcal{S}_1(i) \vdash \mathcal{S}_2(n_2)}{\vdash \mathcal{S}_1(n) \wedge \mathcal{S}_2(n)}\ ccut$$

Since the first premise holds by Lemma 3.1 and the second premise holds by Lemma 3.2, we obtain by the circular cut rule *ccut* and universal generalization that $\mathcal{S}(n) = \mathcal{S}_1(n) \wedge \mathcal{S}_2(n)$ holds for all $n \in \mathbb{N}$. □

The advantage of the circular cut rule in this simple example seems to be negligible, since the same result can be achieved by a simple inductive proof. However, when proving properties of more complex systems with nested circular dependencies, the inductive proof becomes more sophisticated. The application of the circular cut rule, on the other hand, remains as simple as presented here and can in principle be applied automatically. In Chapter 4, many possibilities for applying the circular cut rule in order to prove much more complex properties will appear.

3.4 Compositional Reasoning

Proving properties of real-world systems is rarely possible to be done directly, since such systems are often too large to be handled at once. Thus, several techniques have been investigated to overcome this problem, e.g., symbolic representations, abstraction, and compositional reasoning. Our focus in this section lies on the problem of solving circular dependencies in compositional reasoning, where compositional reasoning means to deduce properties of a composed system from properties of its components.

There are two main applications in practice for compositional reasoning: The first one, as already mentioned, is that the whole system is too large to be handled at once. In this case, the system has to be broken down into small loosely coupled parts (i.e., modules) that can in principle be checked separately. The second one is that we want to infer properties of a system that consists of modules from which we only know their specification but not their internal structure, i.e., we are not able to prove properties of the composed system directly even if it is small enough. In both cases, the aim of compositional reasoning is to deduce properties of the whole system by only using the properties of its components and their dependencies.

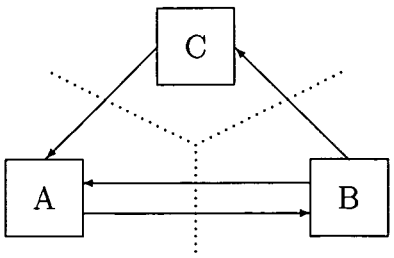


Figure 3.4: Circular composition of modules A, B, and C

This, however, is a very challenging problem in the presence of circular dependencies. Figure 3.4 shows a simple example of such a system.

Note that because of the dependencies between the modules, their properties cannot be proved completely independently. Therefore, when verifying that a particular module guarantees some property, it is in general necessary to make assumptions on other modules. Such kind of reasoning is called *assume-guarantee reasoning*. To get an idea of assume-guarantee reasoning, let us consider the sequential composition rule in Hoare logic [Hoa69]:

$$\frac{\{\mathcal{P}\}\,\Pi_1\,\{\mathcal{R}\} \quad \{\mathcal{R}\}\,\Pi_2\,\{\mathcal{Q}\}}{\{\mathcal{P}\}\,\Pi_1;\Pi_2\,\{\mathcal{Q}\}}$$

The meaning of this rule in the context of assume-guarantee reasoning can be interpreted as: If Π_1 guarantees $\mathcal{R}$ under the assumption $\mathcal{P}$ and Π_2 guarantees $\mathcal{Q}$ under the assumption $\mathcal{R}$, then the sequential composition $\Pi_1;\Pi_2$ of Π_1 and Π_2 guarantees $\mathcal{Q}$ under the assumption $\mathcal{P}$. Note, however, that this interpretation gives only an intuition of assume-guarantee reasoning, but it is not exactly the same. In particular, formulas in assume-guarantee reasoning as proposed by Pnueli [Pnu85] have the form $\langle\varphi\rangle\,\mathfrak{X}\,\langle\psi\rangle$. Although this formula looks like a Hoare triple, it has a different semantics.

Let us write $\mathfrak{X} \sqsubseteq \mathfrak{X}'$ to denote that $\mathfrak{X}$ is a component of $\mathfrak{X}'$ for all systems $\mathfrak{X}$ and $\mathfrak{X}'$.[1] In particular, we require that this relation is transitive and that it satisfies $\mathfrak{X} \sqsubseteq \mathfrak{X}\|\mathfrak{X}'$ for all systems $\mathfrak{X}$ and $\mathfrak{X}'$, where $\mathfrak{X}\|\mathfrak{X}'$ denotes the parallel composition of $\mathfrak{X}$ and $\mathfrak{X}'$.

Definition 3.2. The formula $\langle\varphi\rangle\,\mathfrak{X}\,\langle\psi\rangle$ holds iff whenever $\mathfrak{X} \sqsubseteq \mathfrak{X}'$ such that $\mathfrak{X}'$ satisfies the assumption φ, then $\mathfrak{X}'$ guarantees property ψ.

[1] The exact definition of the relation $\sqsubseteq$ depends on the application area (cf. [GL94]).

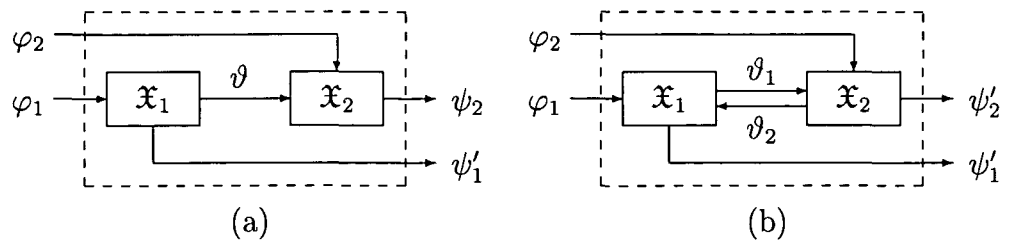


Figure 3.5: Illustration of composition rules

It follows immediately from this definition that for any two systems $\mathfrak{X}$ and $\mathfrak{X}'$ such that $\mathfrak{X} \sqsubseteq \mathfrak{X}'$, it holds that $\langle\varphi\rangle\,\mathfrak{X}\,\langle\psi\rangle$ implies $\langle\varphi\rangle\,\mathfrak{X}'\,\langle\psi\rangle$.

With such formulas at hand, it is now possible to express assume-guarantee inference rules. Such a non-circular composition rule is given by:

$$\frac{\langle\varphi_1\rangle\,\mathfrak{X}_1\,\langle\vartheta \wedge \psi_1\rangle \quad \langle\varphi_2 \wedge \vartheta\rangle\,\mathfrak{X}_2\,\langle\psi_2\rangle}{\langle\varphi_1 \wedge \varphi_2\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\vartheta \wedge \psi_1 \wedge \psi_2\rangle} \tag{3.1}$$

The intuitive meaning of this rule is that if $\mathfrak{X}_1$ guarantees $\vartheta \wedge \psi_1$ under the environment assumption φ_1 and $\mathfrak{X}_2$ guarantees ψ_2 under the environment assumption φ_2 and the assumption ϑ on $\mathfrak{X}_1$, then the composition $\mathfrak{X}_1\|\mathfrak{X}_2$ of $\mathfrak{X}_1$ and $\mathfrak{X}_2$ guarantees $\vartheta \wedge \psi_1 \wedge \psi_2$ under the environment assumption $\varphi_1 \wedge \varphi_2$. An illustration of rule (3.1) is shown in Figure 3.5 (a), where $\psi_1' = \vartheta \wedge \psi_1$. Note that the edges in Figure 3.5 represent the dependencies between the components and their environment and do not indicate any timing behavior.

Theorem 3.3. *The non-circular composition rule (3.1) is sound.*

Proof. We show that the conclusion of rule (3.1) follows from the premises. Since $\mathfrak{X}_2 \sqsubseteq \mathfrak{X}_1\|\mathfrak{X}_2$ and $\langle\varphi_2 \wedge \vartheta\rangle\,\mathfrak{X}_2\,\langle\psi_2\rangle$, we obtain $\langle\varphi_2 \wedge \vartheta\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\psi_2\rangle$, which trivially implies $\langle\varphi_1 \wedge \varphi_2 \wedge \vartheta\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\psi_2\rangle$. Moreover, since $\mathfrak{X}_1 \sqsubseteq \mathfrak{X}_1\|\mathfrak{X}_2$ and $\langle\varphi_1\rangle\,\mathfrak{X}_1\,\langle\vartheta \wedge \psi_1\rangle$ implies $\langle\varphi_1\rangle\,\mathfrak{X}_1\,\langle\vartheta\rangle$, we obtain $\langle\varphi_1\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\vartheta\rangle$. Thus, we know that every $\mathfrak{X}$ with $\mathfrak{X}_1\|\mathfrak{X}_2 \sqsubseteq \mathfrak{X}$ guarantees ϑ under the assumption φ_1, which allows us to remove the assumption ϑ from $\langle\varphi_1 \wedge \varphi_2 \wedge \vartheta\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\psi_2\rangle$. Hence, we have $\langle\varphi_1 \wedge \varphi_2\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\psi_2\rangle$.

In addition, since $\mathfrak{X}_1 \sqsubseteq \mathfrak{X}_1\|\mathfrak{X}_2$ and $\langle\varphi_1\rangle\,\mathfrak{X}_1\,\langle\vartheta \wedge \psi_1\rangle$, we have $\langle\varphi_1\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\vartheta \wedge \psi_1\rangle$, which trivially implies $\langle\varphi_1 \wedge \varphi_2\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\vartheta \wedge \psi_1\rangle$. Therefore, by putting $\langle\varphi_1 \wedge \varphi_2\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\psi_2\rangle$ and $\langle\varphi_1 \wedge \varphi_2\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\vartheta \wedge \psi_1\rangle$ together, we obtain $\langle\varphi_1 \wedge \varphi_2\rangle\,\mathfrak{X}_1\|\mathfrak{X}_2\,\langle\vartheta \wedge \psi_1 \wedge \psi_2\rangle$. □

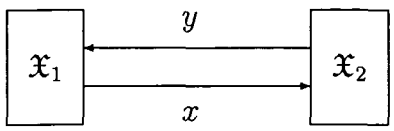


Figure 3.6: Counterexample to circular composition

Note that the composition rule (3.1) is non-circular because $\mathfrak{X}_2$ depends on $\mathfrak{X}_1$ but not vice versa (cf. Figure 3.5 (a)). Our focus, however, lies on the case where $\mathfrak{X}_1$ and $\mathfrak{X}_2$ depend on each other in a circular manner as shown in Figure 3.5 (b). The corresponding circular composition rule, where $\psi_1' = \vartheta_1 \wedge \psi_1$ and $\psi_2' = \vartheta_2 \wedge \psi_2$, is given by:

$$\frac{\langle \varphi_1 \wedge \vartheta_2 \rangle\, \mathfrak{X}_1\, \langle \vartheta_1 \wedge \psi_1 \rangle \quad \langle \varphi_2 \wedge \vartheta_1 \rangle\, \mathfrak{X}_2\, \langle \vartheta_2 \wedge \psi_2 \rangle}{\langle \varphi_1 \wedge \varphi_2 \rangle\, \mathfrak{X}_1 \| \mathfrak{X}_2\, \langle \vartheta_1 \wedge \psi_1 \wedge \vartheta_2 \wedge \psi_2 \rangle} \tag{3.2}$$

In contrast to the non-circular composition rule (3.1) above, the circular composition rule (3.2) is *not* sound in general. Although most counterexamples to this rule in the literature use liveness properties, there exist also counterexamples using safety properties.

Example 3.1. Let $\varphi_1 = \varphi_2 = \top$, $\psi_1 = \psi_2 = \top$, $\vartheta_1 = \mathbf{AG}\,(x = 0)$, and $\vartheta_2 = \mathbf{AG}\,(y = 1)$. Moreover, let $\mathfrak{X}_1 =$ `while y = 1 do x = 0;` and $\mathfrak{X}_2 =$ `while x = 0 do y = 1;`. Figure 3.6 shows a schematic representation of this example. Now, assume that $\mathfrak{X}_1 \| \mathfrak{X}_2$ satisfies $\mathbf{AG}\,(x = 1)$ and $\mathbf{AG}\,(y = 0)$. It is easy to see that this assumption does not contradict the definition of $\mathfrak{X}_1$ and $\mathfrak{X}_2$. Then, both premises

$$\langle \mathbf{AG}\,(y = 1) \rangle \quad \texttt{while y = 1 do x = 0;} \quad \langle \mathbf{AG}\,(x = 0) \rangle$$
$$\langle \mathbf{AG}\,(x = 0) \rangle \quad \texttt{while x = 0 do y = 1;} \quad \langle \mathbf{AG}\,(y = 1) \rangle$$

are true; however, the conclusion of the circular composition rule (3.2) yields

$$\langle \top \rangle\, \mathfrak{X}_1 \| \mathfrak{X}_2\, \langle \mathbf{AG}\,(x = 0) \wedge \mathbf{AG}\,(y = 1) \rangle$$

which contradicts the assumption that $\mathfrak{X}_1 \| \mathfrak{X}_2$ satisfies $\mathbf{AG}\,(x = 1)$ and $\mathbf{AG}\,(y = 0)$. Hence, rule (3.2) is unsound.

Although circular compositions of the form of rule (3.2) are unsound in general, there exist many approaches in the literature to find sufficient

conditions for its soundness. These approaches are typically based on a restriction to a specific formalism and the introduction of techniques to break the circularity. The soundness of the (allegedly) circular rule follows then by an inductive argument.

For example, the circularity in Example 3.1 can be broken by the realistic assumption that the system starts working at a certain point in time at which the variables are initialized with $x = 0$ and $y = 1$. It is then easy to see that there exists no property that is consistent with both $\mathfrak{X}_1$ and $\mathfrak{X}_2$, but contradicts the property inferred by rule (3.2). Thus, in the case of discrete-time systems, the correctness of rule (3.2) can be shown by mutual induction over time, where the initialization of the variables is used as induction start. Note that for real-world systems, such a starting point always exists, e.g., by a call in the case of a software component or by a reset in the case of a hardware component. Therefore, in order to break the circularity, it suffices to ensure that the guaranteed properties of the components are satisfied at the starting point of the system. This holds for all kinds of properties; liveness properties, however, permit a higher degree of freedom to do this, e.g., ensuring that $\mathbf{AF}\,(x = 0)$ holds at the starting point requires that $x = 0$ holds eventually but not necessarily at the starting point itself. Maybe this is the reason why liveness properties are mostly used in the literature as counterexamples to circular composition.

Note that the stepwise propagation between assumptions and guaranteed properties, which allows the application of an induction argument, does not need to progress over (discrete) time. It is easy to see that our approach works over any domain that allows an inductive argument.

The following theorem states the soundness of rule (3.2) if it is restricted in such a way that the circularity is broken as described above. In order to avoid an explicit condition on the starting point of the composed system, we use the same principle as for strong induction (cf. Section 3.3).

Theorem 3.4. *The circular composition rule*

$$\frac{\langle \varphi_1 \wedge \forall i < n_1.\, \vartheta_2(i) \rangle\, \mathfrak{X}_1\, \langle \vartheta_1(n_1) \wedge \psi_1 \rangle \quad \langle \varphi_2 \wedge \forall i < n_2.\, \vartheta_1(i) \rangle\, \mathfrak{X}_2\, \langle \vartheta_2(n_2) \wedge \psi_2 \rangle}{\langle \varphi_1[n_1/n] \wedge \varphi_2[n_2/n] \rangle\, \mathfrak{X}_1 \| \mathfrak{X}_2\, \langle \vartheta_1(n) \wedge \vartheta_2(n) \wedge \psi_1 \wedge \psi_2 \rangle}$$

is sound if φ_1 is downward closed in n_1, φ_2 is downward closed in n_2, ψ_1 does not contain n_1, ψ_2 does not contain n_2, and no formula φ_1, φ_2, ψ_1, ψ_2, ϑ_1, or ϑ_2 contains n.

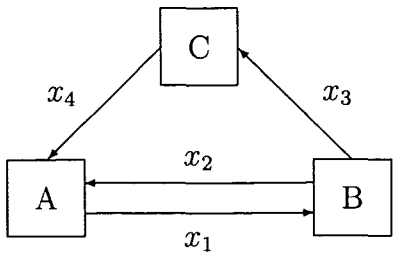


Figure 3.7: Example of circular composition

Proof. Since $\mathfrak{X}_1 \sqsubseteq \mathfrak{X}_1 \| \mathfrak{X}_2$ and $\mathfrak{X}_2 \sqsubseteq \mathfrak{X}_1 \| \mathfrak{X}_2$, we know that:

$$\begin{array}{llll} \langle \varphi_1 \wedge \forall i < n_1 . \vartheta_2(i) \rangle & \mathfrak{X}_1 \| \mathfrak{X}_2 & \langle \vartheta_1(n_1) \wedge \psi_1 \rangle \\ \langle \varphi_2 \wedge \forall i < n_2 . \vartheta_1(i) \rangle & \mathfrak{X}_1 \| \mathfrak{X}_2 & \langle \vartheta_2(n_2) \wedge \psi_2 \rangle \end{array}$$

Thus, in analogy to the circular cut rule (cf. Section 3.2), we obtain the soundness of the circular composition rule. □

The following simple example demonstrates how the circular composition rule can be applied for circular compositional reasoning.

Example 3.2. Let A, B, and C be the modules in Figure 3.7, and suppose that for all $n \in \mathbb{N}$ we have already proved:

$$\langle \forall i < n . x_2(i) \wedge \forall i < n . x_4(i) \rangle \quad A \quad \langle x_1(n) \rangle \tag{3.3}$$

$$\langle \forall i < n . x_1(i) \rangle \quad B \quad \langle x_2(n) \wedge x_3(n) \rangle \tag{3.4}$$

$$\langle \forall i < n . x_3(i) \rangle \quad C \quad \langle x_4(n) \rangle \tag{3.5}$$

We will now show how the circular composition rule can be applied in order to prove that the composed system shown in Figure 3.7 satisfies

$$x_1(n) \wedge x_2(n) \wedge x_3(n) \wedge x_4(n) \text{ for all } n \in \mathbb{N}.$$

Since $\forall i < c + 1 . x_4(i)$ implies $\forall i < c . x_4(i)$ for all $c \in \mathbb{N}$ (i.e., it is downward closed), the circular composition rule is sound according to Theorem 3.4 when applied to the formulas (3.3) and (3.4), which yields

$$\frac{\langle \forall i < n . x_2(i) \wedge \forall i < n . x_4(i) \rangle \, A \, \langle x_1(n) \rangle \quad \langle \forall i < n . x_1(i) \rangle \, B \, \langle x_2(n) \wedge x_3(n) \rangle}{\langle \forall i < n . x_4(i) \rangle \, A \| B \, \langle x_1(n) \wedge x_2(n) \wedge x_3(n) \rangle}$$

Now, by applying the circular composition rule again to the above result and formula (3.5), we obtain

$$\frac{\langle \forall i < n.\, x_4(i) \rangle\ A \| B\ \langle x_1(n) \wedge x_2(n) \wedge x_3(n) \rangle \quad \langle \forall i < n.\, x_3(i) \rangle\ C\ \langle x_4(n) \rangle}{\langle \top \rangle\ A \| B \| C\ \langle x_1(n) \wedge x_2(n) \wedge x_3(n) \wedge x_4(n) \rangle}$$

Thus, since n is a free variable, we can apply universal generalization in order to obtain the desired result. Hence, by successively applying the circular composition rule, we have shown that the composed system in Figure 3.7 satisfies $x_1(n) \wedge x_2(n) \wedge x_3(n) \wedge x_4(n)$ for all $n \in \mathbb{N}$.

3.5 Related Work

The following summarizes the most important publications in circular compositional reasoning and shows how they relate to our work.

Misra and Chandy [MC81] introduced compositional assume-guarantee reasoning for safety properties in the context of process communication. They used assertions of the form $r|h|s$ to specify a process h, where r and s are predicates on traces of communication events. The triple $r|h|s$ denotes that: (i) s holds initially in h, and (ii) if r holds up to the kth point in any trace of h, then s holds up to the $(k+1)$st point in that trace for all $k \geq 0$. It is easy to see that such triples $r|h|s$ implicitly express the conditions of our circular cut rule in order to break circularity. The composition rule is then formulated in their *Theorem of Hierarchy*, which allows to infer properties of process networks from individual processes.

Pnueli [Pnu85] introduced a composition rule that is also able to handle liveness properties. He was the first who proposed the use of temporal logic for writing assume-guarantee specifications, and he introduced assume-guarantee formulas of the form $\langle \varphi \rangle\ \mathfrak{X}\ \langle \psi \rangle$ as we have used in this chapter.

Stark [Sta85] presented an assume-guarantee proof rule independent of the choice of a particular specification or programming language. His composition rule is based on the assumption that there exists a set of specifications that "cuts" the dependencies between the components. Such a *cut set*

must satisfy five conditions: Four conditions on its relation to the assume-guarantee properties and one condition to break circularity. Intuitively, the condition to break circularity requires that on each cycle there is at least one specification in the cut set that holds unconditionally. This property corresponds to the existence of an induction start in our approach.

Abadi and Lamport [AL93, AL95] investigated assume-guarantee reasoning in the case where the assumptions are safety properties but the guaranteed properties can include liveness. In our approach, this means that the guaranteed properties ϑ_1 and ϑ_2 used as assumptions of other components must be safety properties, and the guaranteed properties ψ_1 and ψ_2 that are not used as assumptions can include liveness. Obviously, the liveness properties are then irrelevant concerning circularity, since ψ_1 and ψ_2 do not appear in any cycle. Thus, the remaining properties that are relevant in the context of circular compositions are still safety properties.

The results of Abadi and Lamport are based on semantic arguments and are therefore independent of the choice of a particular specification language or logic. They assume that the actions of the system components are interleaving, i.e., the component's input and output cannot change simultaneously. In particular, this means that a guaranteed property can only become false *after* the corresponding assumption has been violated. Therefore, it is possible to apply an inductive argument when the assumptions are initially satisfied. Hence, the basic reason why their composition rule holds is the same as ours, although their approach is completely different.

McMillan [McM99] generalized the idea of Abadi and Lamport by making the induction over time explicit. In particular, by assuming φ up to time $t-1$ when proving ψ at time t and vice versa, it is possible prove φ and ψ for all $t \in \mathbb{N}$ by mutual strong induction (provided that φ and ψ hold at time 0). Since this approach is independent of the kind of properties expressed by φ and ψ, it extends the results of Abadi and Lamport in the sense that such a circular composition is sound even if the assumptions include liveness properties. McMillan showed how to express this kind of induction in temporal logic in order to verify it by model checking.

It is easy to see that this approach is based on exactly the same principle as our circular cut rule. Our intention, however, was to find a general rule for

solving circular dependencies in inductive proofs not necessarily over time.

Other works. De Roever et al. [dRdBH+01] gave a comprehensive survey on concurrency verification including circular dependencies. Viswanathan and Viswanathan [VV01] investigated a general framework of circular compositional reasoning concerning properties that are expressible by least and greatest fixpoints. At the same time, Maier [Mai01] presented a circular assume-guarantee proof rule within a set-theoretic framework. Most recently, Amla et al. [AENT03] constructed an abstract compositional reasoning framework and defined an assume-guarantee reasoning method that is sound and complete.

3.6 Summary

The classical *cut rule* in logic calculi allows the composition of proofs that depend on each other in the following way: The first proof guarantees property α and the second proof assumes property α. Then, α can be cut out according to the sequent calculus inference rule

$$\frac{\Gamma \vdash \alpha, \Delta \quad \Sigma, \alpha \vdash \Pi}{\Gamma, \Sigma \vdash \Delta, \Pi}\ \mathit{cut}$$

However, if the two proofs depend circularly on each other, i.e., the first proof assumes property β and guarantees property α and the second proof assumes property α and guarantees property β, then the rule

$$\frac{\Gamma, \beta \vdash \alpha, \Delta \quad \Sigma, \alpha \vdash \beta, \Pi}{\Gamma, \Sigma \vdash \Delta, \Pi}$$

is unsound in general. Nevertheless, if α and β are of a certain form, then such a *circular cut rule* can be sound. In particular, the rule is sound if the circularities are of a *spiral* kind, as shown in the following:

$$\frac{\Gamma, \forall i < n_1.\, \beta(i) \vdash \alpha(n_1), \Delta \quad \Sigma, \forall i < n_2.\, \alpha(i) \vdash \beta(n_2), \Pi}{\Gamma[n_1/n], \Sigma[n_2/n] \vdash \alpha(n) \wedge \beta(n), \Delta, \Pi}\ \mathit{ccut}$$

The same principle can be applied in compositional reasoning in order to infer properties of a composed system from properties of its components that depend on each other in a circular manner.

We will use circular compositions very frequently in Chapter 4 in order to obtain our main results from circular dependent auxiliary results. In particular, we will prove properties of query languages which are defined as the union of several sublanguages that depend on each other in a circular manner. To this aim, our strategy will be to prove properties of the sublanguages and compose them afterwards. However, when proving properties of a sublanguage, we have to make assumptions on other sublanguages and vice versa. Nevertheless, by using the principle of our circular cut rule, we will be able to successively remove the assumptions and compose the auxiliary results in order to obtain properties of the whole query languages.

Chapter 4

Exact Temporal Logic Queries

4.1 Introduction

Temporal logic queries (cf. Section 2.5) are a useful formalism for reasoning about temporal logic specifications. They enable us to express incomplete specifications that have to be completed appropriately by instantiating variables in such away that the resulting specifications are satisfied by the model. An interesting case in this context appears when such completions can be reduced to a single completion, i.e., where the set of solutions to a temporal logic query can be characterized by a single solution. Such a strongest solution – if it exists – is called *exact*.

In this chapter, we investigate *exact* temporal logic queries, that are temporal logic queries that always have an exact solution if the set of solutions is not empty. In addition to always having an exact solution provided that there exists any solution, exact temporal logic queries also have other interesting properties. One of the most important ones is *distributivity*. A temporal logic query γ is distributive (over conjunction) iff

$$\gamma[\varphi] \wedge \gamma[\psi] \Leftrightarrow \gamma[\varphi \wedge \psi]$$

for all formulas φ and ψ. The close relationship between exact temporal logic queries and distributivity was already noticed by Chan [Cha00]. In this chapter, we investigate this relationship systematically and show that a temporal logic query is exact if and only if it is distributive.

As conjunction in general distributes over universal quantification only, the distributivity of a query gives evidence that the occurrence of a variable in the query is governed by universal quantification. Moreover, note

that each distributive temporal logic query amounts to a set of equivalences between temporal logic formulas. A simple example is the distributive query $\gamma = \mathbf{G}\,?$, which amounts to the equivalences

$$(\mathbf{G}\,\varphi) \wedge (\mathbf{G}\,\psi) \Leftrightarrow \mathbf{G}(\varphi \wedge \psi)$$

for all formulas φ and ψ. Such simple equivalences were already listed by Emerson [Eme90]. They are obtained as special cases from our results developed in this chapter. In particular, we will present extensive syntactic fragments of exact resp. distributive LTL and CTL queries. In the case of LTL, we are also able to show that our fragment is maximal.

The equivalences between temporal logic formulas based on distributivity over conjunction also have the potential to be exploited in model checking. For example, formulas of the form $\gamma[\varphi \wedge \psi]$ can be split into two smaller formulas $\gamma[\varphi]$ and $\gamma[\psi]$ that can be checked separately. Although it is not to be expected that the reduced formula size will significantly affect the performance of model checking, it is quite likely that each of the smaller formulas ranges over a significantly fewer number of atomic propositions, e.g., by loosely coupled concurrent systems with limited communication and many internal variables. This situation can be exploited in model checking when using existential abstraction, since a smaller number of variables in the specification may reduce the size of the abstract model [CGL94, CGJ+03].

Moreover, the auxiliary results appearing in the exactness proofs of our fragments enable us to solve queries in these fragments efficiently. In particular, several variants of the *collecting* property (i.e., $\gamma[\varphi] \wedge \gamma[\psi] \Rightarrow \gamma[\varphi \wedge \psi]$) will be used in our proofs. These properties can be exploited in order to eliminate existential choices in the evaluation of temporal operators. We will describe this principle in detail in Chapter 5 and present an efficient symbolic query solving algorithm based on this insight.

This chapter is organized as follows: In Section 4.2, we start by a systematic investigation of exact temporal logic queries and present several basic results independently of a particular logic. Afterwards, in Section 4.3, we present a syntactic characterization of exact LTL queries together with the corresponding proofs of exactness and maximality. A syntactic fragment of exact CTL queries together with the corresponding proof of exactness and some comments towards a proof of its maximality are then presented in Section 4.4. Finally, we summarize in Section 4.5.

4.2 Basic Relationships

To obtain the maximum information a query provides, it is necessary to consider *all* its solutions. However, since the number of solutions is likely to be very large, it is desirable to have strongest solutions that subsume all other solutions. We call such strongest solutions *minimal solutions.*[1]

Definition 4.1 (Minimal solution). A set of solutions $\mathcal{M}$ to a query γ in a model $\mathfrak{M}$ is the set of *minimal solutions* iff $\mathcal{M}$ is the smallest set such that for every solution φ to γ in $\mathfrak{M}$ there exists $\mu \in \mathcal{M}$ such that $\mu \Rightarrow \varphi$.

Remark 4.1. Note that the set of minimal solutions does not need to exist in general. For example, because of the infinite implication chain

$$\mathbf{F}p \Leftarrow \mathbf{XF}p \Leftarrow \mathbf{XXF}p \Leftarrow \mathbf{XXXF}p \Leftarrow \cdots,$$

there cannot exist a minimal solution that implies all these solutions in models where p occurs in a cycle. Nevertheless, in order to guarantee the existence of a set of minimal solutions, the formulas that are taken into consideration as solutions can be restricted appropriately. Examples of such restrictions are propositional formulas and length restricted formulas.

It is easy to see that if $\mathcal{M}$ is the set of minimal solutions to γ in $\mathfrak{M}$, then $\mathrm{sol}(\mathfrak{M}, \gamma) \subseteq \{\varphi \mid \exists \mu \in \mathcal{M}.\ \mu \Rightarrow \varphi\}$. The following proposition shows that for monotonic queries also the converse inclusion holds, i.e., every implication of the minimal solutions is a solution. In other words, if γ is monotonic, it holds that $\mathrm{sol}(\mathfrak{M}, \gamma) = \{\varphi \mid \exists \mu \in \mathcal{M}.\ \mu \Rightarrow \varphi\}$.

Lemma 4.1. *Let $\mathcal{M}$ be the set of minimal solutions to a monotonic query γ in a model $\mathfrak{M}$. Then, the formula φ is a solution to γ in $\mathfrak{M}$ iff there exists $\mu \in \mathcal{M}$ such that $\mu \Rightarrow \varphi$.*

Proof. The *only if* direction is essentially the definition of minimal solutions. For the *if* direction, let $\mu \in \mathcal{M}$ such that $\mu \Rightarrow \varphi$. Thus, by monotonicity, we obtain $\gamma[\mu] \Rightarrow \gamma[\varphi]$. Since μ is a solution to γ, it follows that φ must also be a solution. Hence, if $\mu \Rightarrow \varphi$, then φ is a solution to γ in $\mathfrak{M}$. □

[1] Note that the set of solutions to a query together with logical implication form a partially ordered set. The minimal solutions are the minimal elements of this set.

It is easy to see that if the set of minimal solutions exists, then it is semantically unique. However, in the following we are interested in a more specific property; in fact, we are interested in the case where exactly one minimal solution exists. We call this minimal solution the *least solution.*[2]

Definition 4.2 (Least solution). A solution μ to a query γ in a model $\mathfrak{M}$ is the *least solution* iff for every solution φ to γ in $\mathfrak{M}$ it holds that $\mu \Rightarrow \varphi$.

In the following, we define queries that always have a least solution.

Definition 4.3 (Bounded query). A query is *bounded* iff it has a least solution in every model where the set of solutions is not empty.

Remark 4.2. Note that not every monotonic query is bounded. For example, let $\gamma = \mathbf{F}\,?$ be an LTL query and π be a path such that $\ell(\pi(0)) = \{p\}$ and $\ell(\pi(i)) = \{q\}$ for all $i \geq 1$. It is easy to see that γ is monotonic and that p and q are solutions to γ on π, that is $\pi \models \gamma[p] \wedge \gamma[q]$. Now, suppose that there exists a least solution μ to γ on π. Then, we know that $\mu \Rightarrow p$ and $\mu \Rightarrow q$, which is trivially equivalent to $\mu \Rightarrow p \wedge q$. Thus, by Lemma 4.1, it follows that $p \wedge q$ must also be a solution to γ on π. However, it is easy to see that $\pi \not\models \gamma[p \wedge q]$. Hence, γ is monotonic but not bounded.

One the other hand, note also that not every bounded query is monotonic. For example, let $\gamma = ?\,\bar{\mathbf{U}}\,c$ be an LTL query and π be a path such that $\ell(\pi(i)) = \{c, p\}$ for all $i \in \mathbb{N}$. In order to see that γ is bounded, note that the conjunction of all solutions is always a least solution. So it is sufficient to show that $\gamma[\varphi] \wedge \gamma[\psi] \Rightarrow \gamma[\varphi \wedge \psi]$ for all formulas φ and ψ. Suppose that $\sigma \models \gamma[\varphi] \wedge \gamma[\psi]$ for any path σ. By definition, this implies that there exist unique numbers k and l such that $\sigma^{[0,k)} \models \varphi$, $\sigma^{[0,l)} \models \psi$, $\sigma^k \models \neg\varphi \wedge c$, and $\sigma^l \models \neg\psi \wedge c$. Thus, it follows that $\sigma^{[0,\min(k,l))} \models \varphi \wedge \psi$ and $\sigma^{\min(k,l)} \models \neg(\varphi \wedge \psi) \wedge c$, which implies $\sigma \models \gamma[\varphi \wedge \psi]$. So we have shown that γ is bounded. However, it is easy to see that $p \wedge q$ is a solution to γ on π, whereas p is not a solution. Hence, γ is bounded but not monotonic.

Now, we introduce a special solution that exactly characterizes all other solutions to a query. We call such a solution an *exact solution.*

[2]The least solution (if it exists) is the least element in the partially ordered set of solutions with respect to logical implication.

Definition 4.4 (Exact solution). A solution ξ to a query γ in a model $\mathfrak{M}$ is *exact* iff it holds that φ is a solution to γ in $\mathfrak{M}$ iff $\xi \Rightarrow \varphi$.

The following proposition connects monotonic queries, least solutions, and exact solutions. It is directly implied by Lemma 4.1.

Proposition 4.1. *If a monotonic query has a least solution, then it is exact.*

An interesting question in this context is about the relation between the exact solution to a query and the exact solutions to its subqueries. In fact, there is a nice relation when considering the conjunction of two queries.

Proposition 4.2. *Let ξ_1 be the exact solution to query γ_1 and ξ_2 be the exact solution to query γ_2. Then, $\xi_1 \vee \xi_2$ is the exact solution to $\gamma_1 \wedge \gamma_2$.*

Proof. Let ξ_1 and ξ_2 be the exact solutions to γ_1 and γ_2 respectively in model $\mathfrak{M}$. Thus, we have $\mathfrak{M} \models \gamma_1[\varphi]$ iff $\xi_1 \Rightarrow \varphi$ as well as $\mathfrak{M} \models \gamma_2[\varphi]$ iff $\xi_2 \Rightarrow \varphi$. Hence, since $\xi_1 \Rightarrow \xi_1 \vee \xi_2$ and $\xi_2 \Rightarrow \xi_1 \vee \xi_2$, we know that $\mathfrak{M} \models \gamma_1[\xi_1 \vee \xi_2] \wedge \gamma_2[\xi_1 \vee \xi_2]$, which is equivalent to $\mathfrak{M} \models (\gamma_1 \wedge \gamma_2)[\xi_1 \vee \xi_2]$. Thus, $\xi_1 \vee \xi_2$ is a solution to $\gamma_1 \wedge \gamma_2$ in $\mathfrak{M}$. Now, let $(\xi_1 \vee \xi_2) \Rightarrow \varphi$ for any formula φ. This is trivially equivalent to $(\neg\xi_1 \wedge \neg\xi_2) \vee \varphi$, which in turn is equivalent to $(\neg\xi_1 \vee \varphi) \wedge (\neg\xi_2 \vee \varphi)$, that is $(\xi_1 \Rightarrow \varphi) \wedge (\xi_2 \Rightarrow \varphi)$. Since ξ_1 and ξ_2 are exact, we know that this holds iff $\mathfrak{M} \models \gamma_1[\varphi] \wedge \gamma_2[\varphi]$, which is equivalent to $\mathfrak{M} \models (\gamma_1 \wedge \gamma_2)[\varphi]$. Hence, φ is a solution to $\gamma_1 \wedge \gamma_2$ in $\mathfrak{M}$ iff $(\xi_1 \vee \xi_2) \Rightarrow \varphi$. Thus, we have $\mathfrak{M} \models (\gamma_1 \wedge \gamma_2)[\xi_1 \vee \xi_2]$ and $\mathfrak{M} \models (\gamma_1 \wedge \gamma_2)[\varphi]$ iff $(\xi_1 \vee \xi_2) \Rightarrow \varphi$, i.e., $\xi_1 \vee \xi_2$ is an exact solution to $\gamma_1 \wedge \gamma_2$ in $\mathfrak{M}$. □

Remark 4.3. Note that the above result holds also for the special case of the conjunction between a formula and a query, that is $\varphi \wedge \gamma$. Then, the "exact solution" to φ – provided that φ holds in the model – is $\bot$. Hence, if ξ is the exact solution to γ, we obtain $\bot \vee \xi \Leftrightarrow \xi$ as the exact solution to $\varphi \wedge \gamma$, which is obviously true.

The following definition introduces the kind of queries in which we are primarily interested in this chapter. Their set of solutions – provided that it is not empty – can always be exactly characterized by a single formula.

Definition 4.5 (Exact query). A query is *exact* iff it has an exact solution in every model where the set of solutions is not empty.

By putting the above results together, we obtain the equivalence between exact queries and queries that are both bounded and monotonic.

Theorem 4.1. *A query is exact iff it is bounded and monotonic.*

Proof. For the *if* direction, let γ be a bounded and monotonic query. Thus, since γ is bounded, we know that there always exists a least solution if the set of solutions is not empty. Hence, by Proposition 4.1, it follows that γ is exact. For the *only if* direction, let γ be an exact query. Thus, we know that there always exists an exact solution if the set of solutions is not empty. It is easy to see that every exact solution is also a least solution. Hence, γ is bounded. Now, let ξ be the exact solution to γ in any model $\mathfrak{M}$ where the set of solutions is not empty. Suppose that $\varphi \Rightarrow \psi$ and $\mathfrak{M} \models \gamma[\varphi]$, i.e., φ is a solution to γ in $\mathfrak{M}$. Since ξ is exact, we know that $\xi \Rightarrow \varphi$ and therefore $\xi \Rightarrow \psi$. Thus, ψ must also be a solution to γ in $\mathfrak{M}$, that is $\mathfrak{M} \models \gamma[\psi]$. Hence, γ is monotonic. □

Remark 4.4. Note that the general results of Chan [Cha00] are based on the assumption that the considered queries are monotonic. So there is nothing said about the existence of exact queries beyond the class of monotonic queries. The above result, however, implies that there exist no exact queries that are not monotonic. To our best knowledge, this is the first time that this question has been clarified.

In order to present an alternative characterization of exact queries, we need some additional properties.

Definition 4.6 (Collecting, Separating, Distributive). A query γ is *collecting* iff it satisfies $\gamma[\varphi] \wedge \gamma[\psi] \Rightarrow \gamma[\varphi \wedge \psi]$, and *separating* iff it satisfies $\gamma[\varphi \wedge \psi] \Rightarrow \gamma[\varphi] \wedge \gamma[\psi]$ for all formulas φ and ψ. A query is *distributive (over conjunction)* iff it is collecting and separating.

Now, we will show some relations between the above properties. Let us start with a relation between collecting and bounded queries.

Lemma 4.2. *Every collecting query is bounded.*

Proof. Let γ be a collecting query and $\mathfrak{M}$ be a model. Consider the set $\mathcal{S}$ of all solutions to γ in $\mathfrak{M}$, that is $\mathfrak{M} \models \bigwedge_{\varphi \in \mathcal{S}} \gamma[\varphi]$. Thus, since γ is collecting, it follows that $\mathfrak{M} \models \gamma[\bigwedge \mathcal{S}]$. Moreover, since $\bigwedge \mathcal{S} \Rightarrow \varphi$ for every $\varphi \in \mathcal{S}$, we know that $\bigwedge \mathcal{S}$ is a least solution to γ in $\mathfrak{M}$. Hence, γ is bounded. □

The below result states that monotonicity and separability are equivalent.

Lemma 4.3. *A query is separating iff it is monotonic.*

Proof. Let γ be a query. For the *if* direction, consider the valid implications $\varphi \wedge \psi \Rightarrow \varphi$ as well as $\varphi \wedge \psi \Rightarrow \psi$. Thus, by monotonicity, we obtain $\gamma[\varphi \wedge \psi] \Rightarrow \gamma[\varphi]$ as well as $\gamma[\varphi \wedge \psi] \Rightarrow \gamma[\psi]$, which is trivially equivalent to $\gamma[\varphi \wedge \psi] \Rightarrow \gamma[\varphi] \wedge \gamma[\psi]$. Hence, γ is separating. For the *only if* direction, suppose that $\varphi \Rightarrow \psi$. Since $\varphi \Rightarrow \psi$ implies $\varphi \Leftrightarrow \varphi \wedge \psi$, we know that $\gamma[\varphi]$ is equivalent to $\gamma[\varphi \wedge \psi]$. Thus, since γ is separating, we obtain $\gamma[\varphi] \Leftrightarrow \gamma[\varphi \wedge \psi] \Rightarrow \gamma[\psi]$. Hence, γ is monotonic. □

Finally, let us consider the relation between exact and collecting queries.

Lemma 4.4. *Every exact query is collecting.*

Proof. Let γ be an exact query and ξ be its exact solution in a model $\mathfrak{M}$ where the set of solutions is not empty. Further, let φ and ψ be any solutions to γ in $\mathfrak{M}$, that is $\mathfrak{M} \models \gamma[\varphi] \wedge \gamma[\psi]$. By definition, we know that $\xi \Rightarrow \varphi$ and $\xi \Rightarrow \psi$, which is trivially equivalent to $\xi \Rightarrow \varphi \wedge \psi$. Thus, $\varphi \wedge \psi$ must be a solution to γ in $\mathfrak{M}$, that is $\mathfrak{M} \models \gamma[\varphi \wedge \psi]$. Hence, γ is collecting. □

By putting the above auxiliary results together, we obtain an alternative characterization of exact queries in the following theorem.

Theorem 4.2. *A query is exact iff it is distributive.*

Proof. For the *if* direction, we know by Lemma 4.2 and Lemma 4.3 that every distributive query is bounded and monotonic. Hence, by Theorem 4.1, it follows that every distributive query is exact. For the *only if* direction, we know by Theorem 4.1 and Lemma 4.3 that every exact query is separating. In addition, by Lemma 4.4 we know that every exact query is collecting. Hence, every exact query is distributive. □

The following result is directly implied by the previous theorem and the equivalence between separability and monotonicity according to Lemma 4.3.

Corollary 4.1. *A query is exact iff it is monotonic and collecting.*

Remark 4.5. Note that this equivalence provides a naive algorithm for computing exact solutions to exact queries by building the conjunction of all solutions. Due to the collecting property, we know that the resulting formula must be a solution which trivially is a least solution. Thus, by Proposition 4.1, we know that it is also an exact solution.

The following theorem shows the complexity for determining whether a given LTL and CTL query is exact. Since such complexity results depend on the particular formalism, we cannot prove them abstractly. In particular, we use a reduction from and to the validity of LTL and CTL formulas following the proof for validity of CTL queries by Chan [Cha00].

Note that, in contrast to Chan, we use the standard temporal operators in these reductions. Moreover, note that the placeholder occurs only once in the used queries. Hence, the complexity remains unchanged when restricting the query language to the standard temporal operators and by allowing only a single occurrence of the placeholder.

Theorem 4.3. *Deciding whether a given LTL query is exact is* PSPACE*-complete and whether a given CTL query is exact is* EXPTIME*-complete.*

Proof. We show that deciding exactness of LTL and CTL queries is equivalent to deciding validity of LTL and CTL formulas respectively. To this aim, it suffices by Theorem 4.2 to show that deciding distributivity of LTL and CTL queries is equivalent to deciding validity of LTL and CTL formulas respectively. Then, we obtain our result by the fact that deciding validity of LTL formulas is PSPACE-complete [SC85] and deciding validity of CTL formulas is EXPTIME-complete [Eme90].

For the reduction to LTL and CTL validity, note that an LTL resp. CTL query γ is distributive iff the formula $\gamma[p] \wedge \gamma[q] \leftrightarrow \gamma[p \wedge q]$ is valid for some atomic propositions p and q not occurring in γ. The *if* direction follows by contraposition when assuming that γ is not distributive, i.e., there exists a Kripke structure $\mathfrak{K}$ and formulas φ and ψ such that $\mathfrak{K} \not\models \gamma[\varphi] \wedge \gamma[\psi] \leftrightarrow \gamma[\varphi \wedge \psi]$. Now, we construct a new Kripke structure $\mathfrak{K}'$ from $\mathfrak{K}$ by labeling states with new atomic propositions p and q iff φ and ψ hold at these states respectively. Then, it obviously holds that $\mathfrak{K}' \not\models \gamma[p] \wedge \gamma[q] \leftrightarrow \gamma[p \wedge q]$. The *only if* direction follows trivially from the definition of distributivity.

For the reduction from LTL and CTL validity, let φ be an LTL formula and ψ be a CTL formula. Moreover, let p and q be atomic propositions

not occurring in φ or ψ, and let $\gamma_1 = (\mathbf{G}\,\varphi) \vee (p\,\mathbf{U}\,?)$ be an LTL query and $\gamma_2 = (\mathbf{AG}\,\psi) \vee \mathbf{A}(p\,\mathbf{U}\,?)$ be a CTL query. Now, we will show that φ and ψ are valid iff γ_1 and γ_2 are distributive respectively.

The *if* direction follows by contraposition when assuming that φ and ψ are not valid, i.e., there exist Kripke structures $\mathfrak{K}_1$ and $\mathfrak{K}_2$ such that $\mathfrak{K}_1 \not\models \varphi$ and $\mathfrak{K}_2 \not\models \psi$. Since p and q are not occurring in φ or ψ, we can assume w.l.o.g. that the initial states of $\mathfrak{K}_1$ and $\mathfrak{K}_2$ are labeled with p, the immediate successor states of these initial states are labeled with q, and no other states of $\mathfrak{K}_1$ and $\mathfrak{K}_2$ are labeled with p or q. Moreover, we can assume w.l.o.g. that the initial states are not immediate successor states of themselves. Then, it is easy to see that $\mathfrak{K}_1 \models \gamma_1[p] \wedge \gamma_1[q]$ and $\mathfrak{K}_2 \models \gamma_2[p] \wedge \gamma_2[q]$, but $\mathfrak{K}_1 \not\models \gamma_1[p \wedge q]$ and $\mathfrak{K}_2 \not\models \gamma_2[p \wedge q]$. Hence, both queries γ_1 and γ_2 are not collecting and therefore not distributive.

For the *only if* direction, we know that φ and ψ are valid, which implies that also $\mathbf{G}\,\varphi$ and $\mathbf{AG}\,\psi$ are valid. Thus, it is easy to see that $\mathfrak{K} \models \gamma_1[\bot]$ and $\mathfrak{K} \models \gamma_2[\bot]$ for all Kripke structures $\mathfrak{K}$. Hence, by Lemma 2.1, it follows trivially that γ_1 and γ_2 are distributive. □

Therefore, since determining whether a given LTL and CTL query is exact is very hard, we define syntactic fragments of exact LTL and CTL queries in the remainder of this chapter. In particular, in the case of LTL, we are able to present a syntactic characterization. Since the existence of a simple grammar that characterizes all exact LTL queries is improbable according to Theorem 4.3, we use *query templates* in the definition of our grammar. Thus, when proving that a template satisfies some property, the property has to be proved for all instantiations of the template, and when proving that a template does not satisfy some property, the absence of the property has to be proved for a single instantiation of the template. Of course, template characterizations do not capture all exact queries, but they allow an approximation that is consistent with our complexity results.

4.3 Exact LTL Queries

In this section, we present LTLQ^x, an exact query language based on LTL. For simplicity, we restrict our considerations in the following to queries with a single occurrence of the placeholder as introduced by Chan [Cha00]. Note

that this restriction can be slightly weakened by Proposition 4.2 and the additional temporal operators introduced in Section 2.5.1.

Let us start with a class of monotonic LTL queries. To this aim, recall our observations on the monotonicity of temporal operators in Section 2.5.1. It is then easy to define a class of monotonic LTL queries.

Definition 4.7 (LTLQ^m). The language *LTLQ*^{m} is the largest set of LTL queries with a single occurrence of the placeholder that do not contain a subquery of the form $\gamma\,\bar{\mathbf{U}}\,\varphi$ or $\gamma\,\bar{\mathbf{W}}\,\varphi$.

For example, the LTL query $\mathbf{X}\,(\varphi\,\bar{\mathbf{U}}\,\mathbf{G}\,?)$ is in LTLQ^m, whereas $\mathbf{X}\,((\varphi \vee ?)\,\bar{\mathbf{U}}\,\psi)$ is not in LTLQ^m. The following lemma justifies our claim from above and follows directly from Lemma 2.2 and the monotonicity of temporal operators as investigated in Section 2.5.1.

Lemma 4.5. *Every query in LTLQ*^{m} *is monotonic.*

Now, according to Corollary 4.1 and Lemma 4.5, our next step towards an exact query language is to restrict LTLQ^m to a class of collecting queries. To this aim, LTLQ^m must be divided into sublanguages. The corresponding deterministic grammar is shown in Table 4.1, where $\star$ is a special wildcard symbol representing any LTL formula. For example, the template $\star\,\mathbf{U}\,\gamma$ represents all LTL queries of the form $\varphi\,\mathbf{U}\,\gamma$, where φ is an LTL formula.[3]

In the following, we write LTLQ^1 for the language derived from the nonterminal $\langle Q1\rangle$, LTLQ^2 for the language derived from the nonterminal $\langle Q2\rangle$, and so on. It is easy to see that LTLQ$^m = \bigcup_{i=1}^{7}$ LTLQ^i since every operator allowed in LTLQ^m occurs in combination with every non-terminal.

Definition 4.8 (LTLQ^x). The language *LTLQ*^{x} is defined as LTLQ^x = LTLQ$^1 \cup$ LTLQ$^2 \cup$ LTLQ^7. Its complement within LTLQ^m is given by $\overline{\text{LTLQ}^x}$ = LTLQ$^m \backslash$ LTLQ^x = LTLQ$^3 \cup$ LTLQ$^4 \cup$ LTLQ$^5 \cup$ LTLQ^6.

Remark 4.6. Note that all these languages are sets of templates. However, for simplicity we identify these template sets with the sets of queries obtained by instantiating the templates appropriately.

Moreover, note that negation does not appear in the grammar defined in Table 4.1. This, however, is no restriction since after transforming a query

[3] A similar template characterization was also used by Buccafurri et al. [BEGL01].

$\langle Q1 \rangle$	::=	$?$	$\star \wedge \langle Q2 \rangle$	$\langle Q2 \rangle \, \mathring{\mathbf{U}} \star$	$\star \, \mathring{\mathbf{U}} \langle Q2 \rangle$
		$\star \, \bar{\mathbf{U}} \langle Q2 \rangle$	$\langle Q2 \rangle \, \mathring{\mathbf{W}} \star$	$\star \, \mathring{\mathbf{W}} \langle Q2 \rangle$	$\star \, \bar{\mathbf{W}} \langle Q2 \rangle$
		$\star \wedge \langle Q1 \rangle$	$\star \vee \langle Q1 \rangle$	$\mathbf{X} \langle Q1 \rangle$	$\langle Q1 \rangle \, \mathring{\mathbf{U}} \star$
		$\star \, \bar{\mathbf{U}} \langle Q1 \rangle$	$\langle Q1 \rangle \, \mathring{\mathbf{W}} \star$	$\star \, \bar{\mathbf{W}} \langle Q1 \rangle$;	
$\langle Q2 \rangle$	::=	$\langle Q1 \rangle \, \mathbf{U} \star$	$\langle Q1 \rangle \, \mathbf{W} \star$	$\star \vee \langle Q2 \rangle$	$\mathbf{X} \langle Q2 \rangle$
		$\langle Q2 \rangle \, \mathbf{U} \star$	$\star \, \mathbf{U} \langle Q2 \rangle$	$\langle Q2 \rangle \, \mathbf{W} \star$	$\star \, \mathbf{W} \langle Q2 \rangle$;
$\langle Q3 \rangle$	::=	$\mathbf{F} \langle Q1 \rangle$	$\mathbf{F} \langle Q5 \rangle$	$\mathbf{F} \langle Q6 \rangle$	$\mathbf{G} \langle Q4 \rangle$
		$\mathbf{G} \langle Q6 \rangle$	$\langle Q4 \rangle \, \mathring{\mathbf{U}} \star$	$\star \, \mathbf{U} \langle Q1 \rangle$	$\star \, \mathbf{U} \langle Q5 \rangle$
		$\star \, \mathbf{U} \langle Q6 \rangle$	$\star \, \mathring{\mathbf{U}} \langle Q1 \rangle$	$\star \, \mathring{\mathbf{U}} \langle Q5 \rangle$	$\star \, \mathring{\mathbf{U}} \langle Q6 \rangle$
		$\star \, \bar{\mathbf{U}} \langle Q4 \rangle$	$\star \, \bar{\mathbf{U}} \langle Q5 \rangle$	$\star \, \bar{\mathbf{U}} \langle Q6 \rangle$	$\langle Q4 \rangle \, \mathbf{W} \star$
		$\langle Q6 \rangle \, \mathbf{W} \star$	$\langle Q4 \rangle \, \mathring{\mathbf{W}} \star$	$\star \, \mathbf{W} \langle Q5 \rangle$	$\star \, \mathbf{W} \langle Q6 \rangle$
		$\star \, \bar{\mathbf{W}} \langle Q4 \rangle$	$\star \, \bar{\mathbf{W}} \langle Q5 \rangle$	$\star \, \bar{\mathbf{W}} \langle Q6 \rangle$	$\star \wedge \langle Q3 \rangle$
		$\star \vee \langle Q3 \rangle$	$\mathbf{X} \langle Q3 \rangle$	$\mathbf{F} \langle Q3 \rangle$	$\mathbf{G} \langle Q3 \rangle$
		$\langle Q3 \rangle \, \mathring{\mathbf{U}} \star$	$\star \, \mathbf{U} \langle Q3 \rangle$	$\star \, \mathring{\mathbf{U}} \langle Q3 \rangle$	$\star \, \bar{\mathbf{U}} \langle Q3 \rangle$
		$\langle Q3 \rangle \, \mathbf{W} \star$	$\langle Q3 \rangle \, \mathring{\mathbf{W}} \star$	$\star \, \mathbf{W} \langle Q3 \rangle$	$\star \, \bar{\mathbf{W}} \langle Q3 \rangle$;
$\langle Q4 \rangle$	::=	$\langle Q3 \rangle \, \mathbf{U} \star$	$\langle Q5 \rangle \, \mathbf{U} \star$	$\langle Q6 \rangle \, \mathbf{U} \star$	$\langle Q5 \rangle \, \mathbf{W} \star$
		$\star \vee \langle Q4 \rangle$	$\mathbf{X} \langle Q4 \rangle$	$\langle Q4 \rangle \, \mathbf{U} \star$	$\star \, \mathbf{U} \langle Q4 \rangle$
		$\star \, \mathbf{W} \langle Q4 \rangle$;			
$\langle Q5 \rangle$	::=	$\star \, \mathring{\mathbf{U}} \langle Q4 \rangle$	$\star \, \mathbf{W} \langle Q1 \rangle$	$\star \, \mathring{\mathbf{W}} \langle Q1 \rangle$	$\star \, \mathring{\mathbf{W}} \langle Q3 \rangle$
		$\star \, \mathring{\mathbf{W}} \langle Q4 \rangle$	$\star \, \mathring{\mathbf{W}} \langle Q6 \rangle$	$\star \wedge \langle Q5 \rangle$	$\mathbf{X} \langle Q5 \rangle$
		$\langle Q5 \rangle \, \mathring{\mathbf{U}} \star$	$\langle Q5 \rangle \, \mathring{\mathbf{W}} \star$	$\star \, \mathring{\mathbf{W}} \langle Q5 \rangle$;	
$\langle Q6 \rangle$	::=	$\star \wedge \langle Q4 \rangle$	$\star \vee \langle Q5 \rangle$	$\star \wedge \langle Q6 \rangle$	$\star \vee \langle Q6 \rangle$
		$\mathbf{X} \langle Q6 \rangle$	$\langle Q6 \rangle \, \mathring{\mathbf{U}} \star$	$\langle Q6 \rangle \, \mathring{\mathbf{W}} \star$;	
$\langle Q7 \rangle$	::=	$\mathbf{F} \langle Q2 \rangle$	$\mathbf{F} \langle Q4 \rangle$	$\mathbf{G} \langle Q1 \rangle$	$\mathbf{G} \langle Q2 \rangle$
		$\mathbf{G} \langle Q5 \rangle$	$\star \wedge \langle Q7 \rangle$	$\star \vee \langle Q7 \rangle$	$\mathbf{X} \langle Q7 \rangle$
		$\mathbf{F} \langle Q7 \rangle$	$\mathbf{G} \langle Q7 \rangle$	$\langle Q7 \rangle \, \mathbf{U} \star$	$\langle Q7 \rangle \, \mathring{\mathbf{U}} \star$
		$\star \, \mathbf{U} \langle Q7 \rangle$	$\star \, \mathring{\mathbf{U}} \langle Q7 \rangle$	$\star \, \bar{\mathbf{U}} \langle Q7 \rangle$	$\langle Q7 \rangle \, \mathbf{W} \star$
		$\langle Q7 \rangle \, \mathring{\mathbf{W}} \star$	$\star \, \mathbf{W} \langle Q7 \rangle$	$\star \, \mathring{\mathbf{W}} \langle Q7 \rangle$	$\star \, \bar{\mathbf{W}} \langle Q7 \rangle$;

Table 4.1: LTLQ^x production rules

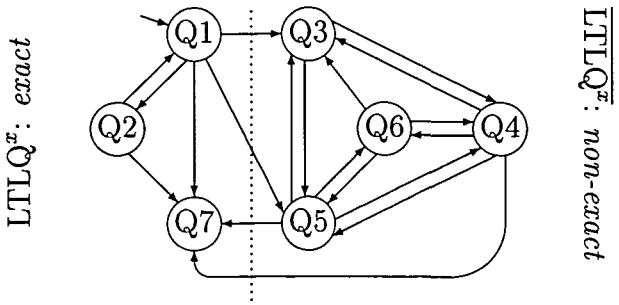


Figure 4.1: LTLQ dependence diagram

into NNF (cf. Section 2.5.1), the negation in front of the placeholder can be removed without loss of generality. To see this, consider a query γ with negation in front of the placeholder. Then, since γ is monotonic decreasing, the exact solution ξ to γ in a model $\mathfrak{M}$ satisfies $\varphi \Rightarrow \xi$ iff φ is a solution to γ in $\mathfrak{M}$. Hence, $\neg\xi \Rightarrow \neg\varphi$ iff $\neg\varphi$ is a solution to γ' in $\mathfrak{M}$, where γ' is obtained from γ by removing the negation in front of the placeholder. Obviously, it holds then that $\neg\xi$ is an exact solution to γ' iff ξ is an exact solution to γ.

The dependencies between the non-terminals in the above grammar are illustrated in Figure 4.1. This graph can be interpreted as an automaton that analyzes a given query starting from the placeholder up to the top-most operator. Its initial state is Q1 because the placeholder occurs in the definition of $\langle Q1 \rangle$. For example, there is a transition from state Q1 to state Q7 because in the definition of non-terminal $\langle Q7 \rangle$ there appears $\mathbf{G}\,\langle Q1 \rangle$, i.e., there is an operator that leads from non-terminal $\langle Q1 \rangle$ to non-terminal $\langle Q7 \rangle$. For simplicity, we omitted the transitions from each state to itself and the labels on each transition. Since the grammar is deterministic, each query can be uniquely assigned to a node in the graph. For example, it can be easily verified that the query $(b\,\mathbf{U}\,(a \wedge ?))\,\mathbf{U}\,c$ belongs to node Q4. The states on the left hand side of the dotted line represent LTLQ^x and the states on the right hand side represent its complement $\overline{\text{LTLQ}^x}$.

In the following, we will show that all instantiations of templates in LTLQ^x are exact and that to each template in $\overline{\text{LTLQ}^x}$ there exists a simple instantiation that is not exact. This will be done by a series of nested inductive proofs on the sublanguages of LTLQ^m. As we shall see soon, a major complication in the proofs arises from the fact that the dependencies between

these sublanguages are circular (cf. Chapter 3). Therefore, the non-trivial dependencies as shown in Figure 4.1 are crucial in our proofs.

Remark 4.7. Note that there exist also exact queries in $\overline{\text{LTLQ}^x}$. However, such queries are instantiations of templates to which also non-exact instantiations exist, i.e., the exactness depends on the chosen instantiation. The existence of a *simple* grammar for *all* exact queries seems to be improbable because it is hard to decide exactness (cf. Theorem 4.3).

4.3.1 Proof of Exactness

This section is devoted to the proof of one of the main results of this thesis, namely the exactness of LTLQ^x. To this aim, we show that all queries in LTLQ^1, LTLQ^2, and LTLQ^7 are collecting. Then, the exactness of LTLQ^x follows by Corollary 4.1. However, since the collecting property introduced in Section 4.2 is too weak, we need the following stronger variants.

Definition 4.9 (Collecting properties). Let γ be an LTL query.
We say γ is *strong collecting* iff for all paths π and formulas φ and ψ:
If $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$, then $\pi^n \models \gamma[\varphi \wedge \psi]$.
We say γ is *boundary collecting* iff for all paths π and formulas φ and ψ:
If $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$, then $\pi^n \models \gamma[\varphi \wedge \psi]$ or $\pi \models \gamma[\bot]$.
We say γ is *intermediate collecting* iff for all paths π and formulas φ and ψ:
If $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$, then $\pi^n \models \gamma[\varphi \wedge \psi]$ or there exists $r < n$ such that $\pi^r \models \gamma[\bot]$.
We say γ is *weak collecting* iff it is collecting.

Note that every strong collecting query is also boundary collecting, every boundary collecting query is also intermediate collecting (consider the case $r = 0$ together with Lemma 2.1), and every intermediate collecting query is also weak collecting (consider the case $n = 0$).

Now, let us start to prove the exactness of LTLQ^x. This will be done by a series of auxiliary results using the above collecting properties.

The following lemma is our first auxiliary result towards a proof for LTLQ^1 and LTLQ^2. Since LTLQ^1 and LTLQ^2 depend on each other (cf. Figure 4.1), we have to make a preliminary assumption on subqueries in LTLQ^1.

Lemma 4.6. *Let $\gamma \in LTLQ^2$. Suppose that every subquery in $LTLQ^1$ is weak collecting. Then, γ is intermediate collecting.*

Proof. Structural induction on γ. See Appendix A.1 for details. □

The following lemma is our second auxiliary result towards a proof for LTLQ^1 and LTLQ^2. Since LTLQ^1 and LTLQ^2 depend on each other (cf. Figure 4.1), we have to make a preliminary assumption on subqueries in LTLQ^2.

Lemma 4.7. *Let $\gamma \in LTLQ^1$. Suppose that every subquery in $LTLQ^2$ is intermediate collecting. Then, γ is weak collecting.*

Proof. Structural induction on γ. See Appendix A.1 for details. □

In order to obtain the assertion of Lemma 4.7 without its assumption, we use an inductive proof on the number of subqueries in LTLQ^2.

Lemma 4.8. *Every query in $LTLQ^1$ is weak collecting.*

Proof. Induction on the number of subqueries in LTLQ^2.
Induction start: If $\gamma \in \mathrm{LTLQ}^1$ contains no subquery in LTLQ^2, then the assumption of Lemma 4.7 is trivially satisfied. Thus, we can apply Lemma 4.7 to γ and obtain that γ is weak collecting.
Induction step: Let $\bar{\gamma}$ be any LTLQ^2 subquery of γ, and $\bar{\bar{\gamma}}$ be any LTLQ^1 subquery of $\bar{\gamma}$. Note that by definition every LTLQ^2 query has an LTLQ^1 subquery. Since the number of LTLQ^2 subqueries of $\bar{\bar{\gamma}}$ must be less than the number of LTLQ^2 subqueries of γ, we can apply the induction hypothesis and obtain that $\bar{\bar{\gamma}}$ is weak collecting. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumption of Lemma 4.6 is satisfied. So we can apply Lemma 4.6 to $\bar{\gamma}$ and obtain that $\bar{\gamma}$ is intermediate collecting. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumption of Lemma 4.7 is satisfied. Hence, we can apply Lemma 4.7 to γ and obtain that γ is weak collecting. □

The following corollary is directly implied by Lemma 4.6 and Lemma 4.8.

Corollary 4.2. *Every query in $LTLQ^2$ is intermediate collecting.*

Remark 4.8. Note that, in principle, Lemma 4.8 and Corollary 4.2 can also be obtained from Lemma 4.6 and Lemma 4.7 by applying our circular cut rule from Section 3.2.

The following lemma is our first auxiliary result towards a proof for LTLQ^7. Since LTLQ^4 and LTLQ^5 depend on each other (cf. Figure 4.1), we have to make a preliminary assumption on subqueries in LTLQ^5.

Lemma 4.9. *Let $\gamma \in LTLQ^4$. Suppose that for every subquery $\bar{\gamma} \in LTLQ^5$ it holds that $\mathbf{G}\,\bar{\gamma}$ is weak collecting. Then, $\mathbf{F}\,\gamma$ is boundary collecting.*

Proof. Structural induction on γ. See Appendix A.1 for details. □

The following lemma is our second auxiliary result towards a proof for LTLQ^7. Since LTLQ^4 and LTLQ^5 depend on each other (cf. Figure 4.1), we have to make a preliminary assumption on subqueries in LTLQ^4.

Lemma 4.10. *Let $\gamma \in LTLQ^5$. Suppose that for every subquery $\bar{\gamma} \in LTLQ^4$ it holds that $\mathbf{F}\,\bar{\gamma}$ is weak collecting. Then, $\mathbf{G}\,\gamma$ is weak collecting.*

Proof. Structural induction on γ. See Appendix A.1 for details. □

In order to obtain the assertion of Lemma 4.9 without its assumption, we use an inductive proof on the number of subqueries in LTLQ^5.

Lemma 4.11. *Let $\gamma \in LTLQ^4$. Then, $\mathbf{F}\,\gamma$ is boundary collecting.*

Proof. Induction on the number of subqueries in LTLQ^5.
Induction start: If γ contains no subquery in LTLQ^5, then the assumption of Lemma 4.9 is trivially satisfied. Hence, we can apply Lemma 4.9 to γ and obtain that $\mathbf{F}\,\gamma$ is boundary collecting.
Induction step: Let $\bar{\gamma}$ be any LTLQ^5 subquery of γ. If $\bar{\gamma}$ contains no LTLQ^4 subquery, then the assumption of Lemma 4.10 is trivially satisfied. Otherwise, let $\bar{\bar{\gamma}}$ be any LTLQ^4 subquery of $\bar{\gamma}$. Since the number of LTLQ^5 subqueries of $\bar{\bar{\gamma}}$ must be less than the number of LTLQ^5 subqueries of γ, we can apply the induction hypothesis and obtain that $\mathbf{F}\,\bar{\bar{\gamma}}$ is boundary collecting. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumption of Lemma 4.10 is again satisfied. So in both cases we can apply Lemma 4.10 to $\bar{\gamma}$ and obtain that $\mathbf{G}\,\bar{\gamma}$ is weak collecting. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumption of Lemma 4.9 is satisfied. Hence, we can apply Lemma 4.9 to γ and obtain that $\mathbf{F}\,\gamma$ is boundary collecting. □

The following corollary is directly implied by Lemma 4.10 and Lemma 4.11.

Corollary 4.3. *Let $\gamma \in LTLQ^5$. Then, $\mathbf{G}\,\gamma$ is strong collecting.*

Proof. Suppose that $\pi \models \mathbf{G}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{G}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Then, it is easy to see that $\pi^n \models \mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$. Hence, since $\mathbf{G}\,\gamma$ is weak collecting by Lemma 4.10 and Lemma 4.11, we obtain $\pi^n \models \mathbf{G}\,\gamma[\varphi \wedge \psi]$. □

Remark 4.9. Note that, in principle, Lemma 4.11 and Corollary 4.3 can also be obtained from Lemma 4.9 and Lemma 4.10 by applying our circular cut rule from Section 3.2.

Note that every query in LTLQ^7 contains a subquery of the form $\mathbf{F}\,\gamma$, where $\gamma \in \mathrm{LTLQ}^2 \cup \mathrm{LTLQ}^4$, or of the form $\mathbf{G}\,\gamma$, where $\gamma \in \mathrm{LTLQ}^1 \cup \mathrm{LTLQ}^2 \cup \mathrm{LTLQ}^5$. Therefore, we can use Lemma 4.8, Corollary 4.2, Lemma 4.11, and Corollary 4.3 as induction start in the proof of the following lemma.

Lemma 4.12. *Every query in LTLQ^7 is boundary collecting.*

Proof. Structural induction on γ. See Appendix A.1 for details. □

Remark 4.10. Note that Lemma 4.11 and Corollary 4.3 are used in the proof of Lemma 4.12. This means that properties of queries that are not exact (i.e., queries in LTLQ^4 and LTLQ^5) are needed in order to prove the exactness of queries in LTLQ^7 (cf. Figure 4.1).

Now, recall that LTLQ^x is defined as the union of LTLQ^1, LTLQ^2, and LTLQ^7 (cf. Definition 4.8). Moreover, as already mentioned, every boundary collecting query is intermediate collecting and every intermediate collecting query is (weak) collecting. Thus, we obtain the following corollary by Lemma 4.8, Corollary 4.2, and Lemma 4.12.

Corollary 4.4. *Every query in LTLQ^x is collecting.*

Hence, since LTLQ^x is a subset of LTLQ^m, we obtain by Lemma 4.5, Corollary 4.4, and Corollary 4.1 one of the main results of this thesis.

Theorem 4.4. *Every query in LTLQ^x is exact.*

4.3.2 Proof of Maximality

In this section, we will show that LTLQ^x is maximal in the sense that each template in $\overline{\mathrm{LTLQ}^x}$ has a simple instantiation that is not collecting and therefore not exact. To this aim, we inductively construct a path π for each such instantiation γ such that $\pi \models \gamma[p] \wedge \gamma[q]$ but $\pi \not\models \gamma[p \wedge q]$. In particular, the instantiations are simple in the following sense.

Definition 4.10 (Simple query). A query γ is *simple* iff every subformula (without placeholder) of γ is atomic and occurs only once in γ. We denote the set of atomic propositions occurring in γ by $\mathrm{aprop}(\gamma)$.

For example, the query $\mathbf{G}\,(a\,\mathbf{U}\,(b \wedge \mathbf{X}\,?))$ is simple whereas the queries $\mathbf{G}\,((a \vee b)\,\mathbf{U}\,\mathbf{X}\,?)$ and $\mathbf{G}\,(a\,\mathbf{U}\,\mathbf{X}\,(a \wedge ?))$ are not simple because $a \vee b$ is not atomic and a occurs twice respectively. Note that this property restricts only the subformulas of a query.

It is easy to see that for every subquery $\bar{\bar{\gamma}}$ of a simple query $\gamma = \bar{\gamma}[\bar{\bar{\gamma}}]$, it holds that $\mathrm{aprop}(\bar{\gamma}) \cap \mathrm{aprop}(\bar{\bar{\gamma}}) = \emptyset$. Thus, all subformulas of a simple query are independent of each other as they are different atomic propositions. This allows the inductive construction of a counterexample path by labeling the states according to the atomic propositions in a query without affecting the truth value of other subformulas.

Example 4.1. Consider the simple query $\gamma = (b\,\mathbf{U}\,(a \wedge ?))\,\mathbf{U}\,c$. Since $\gamma \in \mathrm{LTLQ}^4$, there exists a path π such that $\pi \models \gamma[p] \wedge \gamma[q]$ but $\pi \not\models \gamma[p \wedge q]$ for any atomic propositions p and q not occurring in γ. The construction of π for this simple example according to our proof is illustrated in Figure 4.2. We start with an initial path on which for all $n \in \mathbb{N}$ it holds that $\pi^{4n} \models p$, $\pi^{4n+2} \models q$, and $\pi \models \mathbf{G}\,\neg(p \wedge q)$. Then, according to the structure of γ, we successively add labels to the states of π in such a way that we can always achieve resp. preserve a counterexample by adding further labels. It is easy to see that the resulting path in Figure 4.2 is indeed a counterexample to the collecting property. Note that there exist simpler ad hoc counterexamples to the query γ. The construction in Figure 4.2, however, illustrates the general method that works for all queries.

An operation that we will also need for such a construction is the following.

Definition 4.11 (Concatenation). Let σ be a path prefix of length n and π be a path. Then, the *concatenation* of σ and π is given by

$$(\sigma \circ \pi)(i) = \begin{cases} \sigma(i) & : \; i < n \\ \pi(i) & : \; i \geq n \end{cases} \qquad \text{for all } i \in \mathbb{N}.$$

Note that we will only use the special case where the path prefix σ consists of a single state. For simplicity, we identify this single state with the path prefix itself, i.e., we identify state s with the path prefix $\sigma : 0 \mapsto s$.

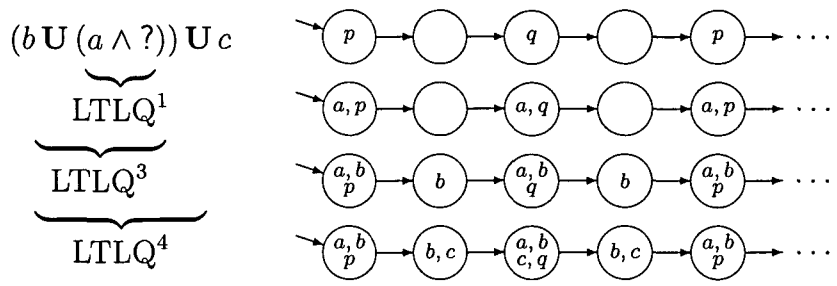


Figure 4.2: Counterexample path construction

Now, let us start with the auxiliary results towards a counterexample path construction for queries in $\overline{\text{LTLQ}^x}$. As in the exactness proof of LTLQ^x, the tricky part in the current proof is to find suitable auxiliary results.

Recall that every query in $\overline{\text{LTLQ}^x}$ has a subquery in LTLQ^1, which will be used as the starting point in our proof. At first, we need the following property, since LTLQ^1 and LTLQ^2 depend on each other (cf. Figure 4.1).

Lemma 4.13. *Let $\gamma \in \mathit{LTLQ}^1 \cup \mathit{LTLQ}^2$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi^{4n} \models \gamma[p]$, $\pi^{4n+2} \models \gamma[q]$, and $\pi^{2n} \not\models \gamma[p \wedge q]$ for all $n \in \mathbb{N}$.*

Proof. Structural induction on γ. See Appendix A.2 for details. □

Since LTLQ^2 queries are the only subqueries of queries in LTLQ^1 (cf. Figure 4.1), the previous lemma can be used to show the following property.

Lemma 4.14. *Let $\gamma \in \mathit{LTLQ}^1$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi^{4n} \models \gamma[p]$ and $\pi^{4n+2} \models \gamma[q]$ for all $n \in \mathbb{N}$ as well as $\pi \models \mathbf{G}\, \neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ. See Appendix A.2 for details. □

Remark 4.11. Note that Lemma 4.14 is used in the proofs of Lemma 4.15 and Lemma 4.17. This means that a property of queries that are exact (in particular, queries in LTLQ^1) is needed in order to prove the non-exactness of queries in LTLQ^3 and LTLQ^5 (cf. Figure 4.1).

Now, we will prove a series of auxiliary results. These results will then be composed in order to obtain a counterexample path to the collecting property for all simple queries in $\overline{\text{LTLQ}^x}$.

The following lemma is an auxiliary result on LTLQ^5 and LTLQ^6. Since $\text{LTLQ}^5 \cup \text{LTLQ}^6$ and $\text{LTLQ}^3 \cup \text{LTLQ}^4$ depend on each other (cf. Figure 4.1), we have to make preliminary assumptions on subqueries in LTLQ^3 and LTLQ^4. For subqueries in LTLQ^1, we can directly use Lemma 4.14.

Lemma 4.15. *Let $\gamma \in LTLQ^5 \cup LTLQ^6$ be simple. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^3$ and $LTLQ^4$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Then, there exists a path π such that $\pi^{4n} \models \gamma[p] \wedge \gamma[q]$ for all $n \in \mathbb{N}$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ. See Appendix A.2 for details. □

The following lemma is an auxiliary result on LTLQ^6 that will only be used in the proof of Lemma 4.17. Because of the additional assumption of a preceding global operator, we obtain a stronger result than in Lemma 4.15.

Lemma 4.16. *Let $\gamma = \mathbf{G}\,\bar{\bar{\gamma}}$ be a simple LTL query where $\bar{\bar{\gamma}} \in LTLQ^6$. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^4$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma \models \mathbf{G}\,\bar{\gamma}[p] \wedge \mathbf{G}\,\bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Suppose further that for every $LTLQ^5$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Then, there exists a path π such that $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ. See Appendix A.2 for details. □

The following lemma is an auxiliary result on LTLQ^3. Since LTLQ^3 and $\text{LTLQ}^4 \cup \text{LTLQ}^5 \cup \text{LTLQ}^6$ depend on each other (cf. Figure 4.1), we have to make preliminary assumptions on subqueries in LTLQ^4, LTLQ^5, and LTLQ^6. For subqueries in LTLQ^1 and subqueries in LTLQ^6 that are preceded by a global operator, we can directly use Lemma 4.14 and Lemma 4.16.

Lemma 4.17. *Let $\gamma \in LTLQ^3$ be simple. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^4$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma \models \mathbf{G}\,\bar{\gamma}[p] \wedge \mathbf{G}\,\bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for*

all $n \in \mathbb{N}$. *Suppose further that for every* $LTLQ^5$ *and* $LTLQ^6$ *subquery* $\bar{\gamma}$ *there exists a path* σ *such that* $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ *for all* $n \in \mathbb{N}$ *and* $\sigma \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. *Then, there exists a path* π *such that* $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ *and* $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.

Proof. Structural induction on γ. See Appendix A.2 for details. □

The following lemma is an auxiliary result on LTLQ^4. Since LTLQ^4 and $\mathrm{LTLQ}^3 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$ depend on each other (cf. Figure 4.1), we have to make preliminary assumptions on subqueries in LTLQ^3, LTLQ^5, and LTLQ^6.

Lemma 4.18. *Let* $\gamma \in LTLQ^4$ *be simple. Further, let* p *and* q *be atomic propositions not occurring in* γ. *Suppose that for every* $LTLQ^3$, $LTLQ^5$, *and* $LTLQ^6$ *subquery* $\bar{\gamma}$ *there exists a path* σ *such that* $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ *and* $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ *for all* $n \in \mathbb{N}$. *Then, there exists a path* π *such that* $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ *and* $\pi^{4n} \not\models \gamma[p \wedge q]$ *for all* $n \in \mathbb{N}$.

Proof. Structural induction on γ. See Appendix A.2 for details. □

Now, we have finished our basic results. In the following, we will successively reduce the number of assumptions in the above lemmas; in particular, we will first remove the assumptions on LTLQ^4 subqueries. In order to obtain the assertion of Lemma 4.17 without its assumption on LTLQ^4 subqueries, we use an inductive proof on the number of subqueries in LTLQ^4.

Lemma 4.19. *Let* $\gamma \in LTLQ^3$ *be simple. Further, let* p *and* q *be atomic propositions not occurring in* γ. *Suppose that for every* $LTLQ^5$ *and* $LTLQ^6$ *subquery* $\bar{\gamma}$ *there exists a path* σ *such that* $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ *for all* $n \in \mathbb{N}$ *and* $\sigma \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. *Then, there exists a path* π *such that* $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ *and* $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.

Proof. Induction on the number of subqueries in LTLQ^4.
Induction start: If γ contains no subquery in LTLQ^4, then the assumption of Lemma 4.17 on subqueries in LTLQ^4 is trivially satisfied. Thus, the only remaining assumptions of Lemma 4.17 are on queries in LTLQ^5 and LTLQ^6, which are satisfied by the actual assumptions. Hence, we obtain the assertion by Lemma 4.17.
Induction step: Let $\bar{\gamma}$ be any LTLQ^4 subquery of γ. If $\bar{\gamma}$ contains no

subquery in LTLQ^3, then the assumption of Lemma 4.18 on subqueries in LTLQ^3 is trivially satisfied. Otherwise, let $\bar{\bar{\gamma}}$ be any LTLQ^3 subquery of $\bar{\gamma}$. Since the number of LTLQ^4 subqueries of $\bar{\bar{\gamma}}$ must be less than the number of LTLQ^4 subqueries of γ, we can apply the induction hypothesis and obtain the assertion for $\bar{\bar{\gamma}}$. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumption of Lemma 4.18 on subqueries in LTLQ^3 is again satisfied. Hence, the only remaining assumptions of Lemma 4.18 in both cases are on queries in LTLQ^5 and LTLQ^6, which are satisfied by the actual assumptions. So in both cases we can apply Lemma 4.18 to $\bar{\gamma}$. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumption of Lemma 4.17 on subqueries in LTLQ^4 is satisfied. Thus, the only remaining assumptions of Lemma 4.17 are on queries in LTLQ^5 and LTLQ^6, which are satisfied by the actual assumptions. Hence, we obtain the assertion by Lemma 4.17. □

In order to obtain the assertion of Lemma 4.15 without its assumption on LTLQ^4 subqueries, we use an inductive proof on the number of subqueries in LTLQ^4.

Lemma 4.20. *Let $\gamma \in \mathit{LTLQ}^5 \cup \mathit{LTLQ}^6$ be simple. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every LTLQ^3 subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Then, there exists a path π such that $\pi^{4n} \models \gamma[p] \wedge \gamma[q]$ for all $n \in \mathbb{N}$ and $\pi \models \mathbf{G}\, \neg\gamma[p \wedge q]$.*

Proof. Induction on the number of subqueries in LTLQ^4.
Induction start: If γ contains no subquery in LTLQ^4, then the assumption of Lemma 4.15 on subqueries in LTLQ^4 is trivially satisfied. Thus, the only remaining assumption of Lemma 4.15 is on queries in LTLQ^3, which is satisfied by the actual assumption. Hence, we obtain the assertion by Lemma 4.15.
Induction step: Let $\bar{\gamma}$ be any LTLQ^4 subquery of γ. If $\bar{\gamma}$ contains no subquery in $\mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$, then the assumptions of Lemma 4.18 on subqueries in LTLQ^5 and LTLQ^6 are trivially satisfied. Otherwise, let $\bar{\bar{\gamma}}$ be any LTLQ^5 or LTLQ^6 subquery of $\bar{\gamma}$. Since the number of LTLQ^4 subqueries of $\bar{\bar{\gamma}}$ must be less than the number of LTLQ^4 subqueries of γ, we can apply the induction hypothesis and obtain the assertion for $\bar{\bar{\gamma}}$. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumptions of Lemma 4.18 on subqueries in LTLQ^5 and LTLQ^6 are again satisfied. Hence, the only remaining assumption of

Lemma 4.18 in both cases is on queries in LTLQ^3, which is satisfied by the actual assumption. So in both cases we can apply Lemma 4.18 to $\bar{\gamma}$. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumption of Lemma 4.15 on subqueries in LTLQ^4 is satisfied. Thus, the only remaining assumption of Lemma 4.15 is on queries in LTLQ^3, which is satisfied by the actual assumption. Hence, we obtain the assertion by Lemma 4.15. □

Now, we have obtained each result with assumptions on at most two kinds of subqueries. In the following we continue in the same manner as above. In order to obtain the assertion of Lemma 4.20 without assumptions, we use an inductive proof on the number of subqueries in LTLQ^3.

Lemma 4.21. *Let $\gamma \in \mathit{LTLQ}^5 \cup \mathit{LTLQ}^6$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi^{4n} \models \gamma[p] \wedge \gamma[q]$ for all $n \in \mathbb{N}$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Induction on the number of subqueries in LTLQ^3.
Induction start: If γ contains no subquery in LTLQ^3, then the assumption of Lemma 4.20 on subqueries in LTLQ^3 is trivially satisfied and we obtain the assertion by Lemma 4.20.
Induction step: Let $\bar{\gamma}$ be any LTLQ^3 subquery of γ. If $\bar{\gamma}$ contains no subquery in $\mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$, then the assumptions of Lemma 4.19 are trivially satisfied. Otherwise, let $\bar{\bar{\gamma}}$ be any LTLQ^5 or LTLQ^6 subquery of $\bar{\gamma}$. Since the number of LTLQ^3 subqueries of $\bar{\bar{\gamma}}$ must be less than the number of LTLQ^3 subqueries of γ, we can apply the induction hypothesis and obtain the assertion for $\bar{\bar{\gamma}}$. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumptions of Lemma 4.19 are again satisfied. So in both cases we can apply Lemma 4.19 to $\bar{\gamma}$. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumption of Lemma 4.20 is satisfied. Hence, we obtain the assertion by Lemma 4.20. □

Since the assumptions of Lemma 4.19 are satisfied according to Lemma 4.21, we trivially obtain the following corollary by Lemma 4.19.

Corollary 4.5. *Let $\gamma \in \mathit{LTLQ}^3$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Since the assumptions of Lemma 4.18 are satisfied according to Lemma 4.21 and Corollary 4.5, we trivially obtain the following corollary by Lemma 4.18.

f_φ^γ	$\varphi = p$	$\varphi = q$	$\varphi = p \wedge q$
$\gamma \in \mathrm{LTLQ}^3$	n	n	n
$\gamma \in \mathrm{LTLQ}^4$	n	n	$4n$
$\gamma \in \mathrm{LTLQ}^5$	$4n$	$4n$	n
$\gamma \in \mathrm{LTLQ}^6$	$4n$	$4n$	n

Table 4.2: Structure of counterexample paths

Corollary 4.6. *Let $\gamma \in LTLQ^4$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ and $\pi^{4n} \not\models \gamma[p \wedge q]$ for all $n \in \mathbb{N}$.*

Remark 4.12. Note that, in principle, Lemma 4.21, Corollary 4.5, and Corollary 4.6 can also be obtained from to Lemma 4.15, Lemma 4.17, and Lemma 4.18 by successively applying our circular cut rule from Section 3.2.

Now, let us summarize our results so far: According to Lemma 4.21, Corollary 4.5, and Corollary 4.6, we know that for each simple query γ in LTLQ^3, LTLQ^4, LTLQ^5, and LTLQ^6 there exists a path π such that

$$\pi^{f_p^\gamma(n)} \models \gamma[p] \quad \text{and} \quad \pi^{f_q^\gamma(n)} \models \gamma[q], \quad \text{but} \quad \pi^{f_{p\wedge q}^\gamma(n)} \not\models \gamma[p \wedge q]$$

where $f_\varphi^\gamma : \mathbb{N} \to \mathbb{N}$ is a function in n defined in Table 4.2. Recall that $\overline{\mathrm{LTLQ}^x}$ is defined as the union of LTLQ^3, LTLQ^4, LTLQ^5, and LTLQ^6 (cf. Definition 4.8). Thus, by Lemma 4.21, Corollary 4.5, and Corollary 4.6 we obtain the following property from the special case of $n = 0$.

Corollary 4.7. *Let $\gamma \in \overline{LTLQ^x}$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi \models \gamma[p] \wedge \gamma[q]$ but $\pi \not\models \gamma[p \wedge q]$, i.e., γ is not collecting.*

Finally, we obtain one of our main results by Corollary 4.7 and Theorem 4.2.

Theorem 4.5. *Every simple query in $\overline{LTLQ^x}$ is not exact.*

Thus, we have shown that LTLQ^x is maximal in the sense that all templates not in LTLQ^x have simple instantiations that are not exact. Hence, LTLQ^x represents a syntactic characterization of exact LTL queries.

Remark 4.13. As already mentioned, $\overline{\text{LTLQ}^x}$ contains also exact queries. However, these queries cannot be simple. In fact, whether a query in $\overline{\text{LTLQ}^x}$ is exact depends on the chosen instantiation. For example, the query $a\,\mathbf{U}\,(b \wedge (?\,\mathbf{U}\,c)) \in \text{LTLQ}^3$ is simple and therefore not exact. In contrast, the queries $a\,\mathbf{U}\,(b \wedge (?\,\mathbf{U}\,\mathbf{G}\,c)) \in \text{LTLQ}^3$ and $((b\,\mathbf{U}\,?) \wedge \mathbf{X}\,\neg a)\,\mathbf{U}\,a \in \underline{\text{LTLQ}^4}$ are exact. Note that it is not the case that all non-simple queries in $\overline{\text{LTLQ}^x}$ are exact. For example, the query $a\,\mathbf{U}\,(\mathbf{X}\,b \wedge (?\,\mathbf{U}\,c)) \in \text{LTLQ}^3$ is not simple and not exact. It certainly would be interesting to have a characterization of *all* exact queries. In consideration of the PSPACE-completeness (cf. Theorem 4.3), however, this seems to be very difficult. Nevertheless, there remains the possibility of a characterization up to logical equivalence as Maidl's characterization of ACTL $\cap$ LTL [Mai00].

4.4 Exact CTL Queries

In this section, we present CTLQ^x, an exact query language based on CTL. As in the case of LTL, we restrict our considerations to queries with a single occurrence of the placeholder as introduced by Chan [Cha00]. Recall that this restriction can be slightly weakened by Proposition 4.2 and the additional temporal operators introduced in Section 2.5.1.

Let us start with a class of monotonic CTL queries. To this aim, recall our observations on the monotonicity of temporal operators in Section 2.5.1. Moreover, note that our aim is to define a fragment of exact CTL queries within the class of monotonic CTL queries. However, it is easy to see that CTL queries containing an existential path quantifier cannot be exact because the existence of solutions φ and ψ on two different paths respectively does not guarantee the existence of a solution $\varphi \wedge \psi$, which contradicts distributivity (cf. Theorem 4.2).

Example 4.2. For example, consider the CTL query $\gamma = \mathbf{EX}\,?$ and a Kripke structure $\mathfrak{K}$ with initial state s_0 consisting of two paths π_1 and π_2 such that $s_0 = \pi_1(0) = \pi_2(0)$, $\ell(\pi_1(1)) = \{p\}$, and $\ell(\pi_2(1)) = \{q\}$. Then, it trivially holds that $\mathfrak{K} \models \gamma[p]$ and $\mathfrak{K} \models \gamma[q]$, but $\mathfrak{K} \not\models \gamma[p \wedge q]$.

Thus, it suffices to consider monotonic ACTL queries. Similarly to the case of LTL, such a class can be easily defined.

Definition 4.12 (ACTLQ^m). The language $ACTLQ^m$ is the largest set of ACTL queries with a single occurrence of the placeholder that do not contain a subquery of the form $\mathbf{A}(\gamma\,\bar{\mathbf{U}}\,\varphi)$ or $\mathbf{A}(\gamma\,\bar{\mathbf{W}}\,\varphi)$.

For example, the CTL query $\mathbf{AX}\,\mathbf{A}(\varphi\,\bar{\mathbf{U}}\,\mathbf{AG}\,?)$ is in ACTLQ^m, whereas $\mathbf{AX}\,\mathbf{A}((\varphi \vee ?)\,\bar{\mathbf{U}}\,\psi)$ is not in ACTLQ^m. The following lemma justifies our claim from above and follows directly from Lemma 2.2 and the monotonicity of temporal operators as investigated in Section 2.5.1.

Lemma 4.22. *Every query in $ACTLQ^m$ is monotonic.*

Now, according to Corollary 4.1 and Lemma 4.5, our next step towards an exact query language is to restrict ACTLQ^m to a class of collecting queries. To this aim, ACTLQ^m must be divided into sublanguages. The corresponding deterministic grammar is shown in Table 4.3 and Table 4.4, where $\star$ is a special wildcard symbol representing any CTL formula. For example, the template $\mathbf{A}(\star\,\mathbf{U}\,\gamma)$ represents all CTL queries of the form $\mathbf{A}(\varphi\,\mathbf{U}\,\gamma)$, where φ is a CTL formula.[4]

In the following, we write CTLQ^1 for the language derived from the nonterminal $\langle Q1\rangle$, CTLQ^2 for the language derived from the nonterminal $\langle Q2\rangle$, and so on. It is easy to see that ACTLQ$^m = \bigcup_{i=1}^{11}$ CTLQ^i since every operator allowed in ACTLQ^m occurs in combination with every non-terminal.

Definition 4.13 (CTLQ^x). The language $CTLQ^x$ is defined as CTLQ$^x = \bigcup_{i=1}^{10}$ CTLQ^i. Its complement within ACTLQ^m is given by $\overline{\text{CTLQ}^x}$ = ACTLQ$^m \backslash$ CTLQ^x = CTLQ^{11}.

Remark 4.14. Note that all these languages are sets of templates. However, for simplicity we identify these template sets with the sets of queries obtained by instantiating the templates appropriately.

Moreover, note that negation does not appear in the grammar defined in Table 4.3 and Table 4.4. This, however, is no restriction since the negation in front of the placeholder can be removed without loss of generality. Since negation in is only allowed in front of atomic propositions and the placeholder in order to avoid implicit existential path quantification, the same argument as in the case of LTLQ^x can be applied.

[4] A similar template characterization was also used by Buccafurri et al. [BEGL01].

$\langle Q1 \rangle$	$::=$	$?$	$\mid$	$\star \wedge \langle Q3 \rangle$	$\mid$	$\star \wedge \langle Q4 \rangle$	$\mid$
		$\star \vee \langle Q2 \rangle$	$\mid$	$\mathbf{AX}\ \langle Q3 \rangle$	$\mid$	$\mathbf{AX}\ \langle Q4 \rangle$	$\mid$
		$\mathbf{AX}\ \langle Q6 \rangle$	$\mid$	$\mathbf{AX}\ \langle Q7 \rangle$	$\mid$	$\mathbf{A}(\ \langle Q3 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \langle Q4 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{U}}\ \langle Q4 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{U}}\ \langle Q5 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q2 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q3 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q4 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q5 \rangle\ )$	$\mid$	$\mathbf{A}(\ \langle Q3 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q4 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q2 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q3 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q4 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q5 \rangle\ )$	$\mid$	$\star \wedge \langle Q1 \rangle$	$\mid$	$\star \vee \langle Q1 \rangle$	$\mid$
		$\mathbf{AX}\ \langle Q1 \rangle$	$\mid$	$\mathbf{A}(\ \langle Q1 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q1 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \langle Q1 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q1 \rangle\ )$	;		
$\langle Q2 \rangle$	$::=$	$\star \wedge \langle Q5 \rangle$	$\mid$	$\mathbf{AX}\ \langle Q5 \rangle$	$\mid$	$\mathbf{A}(\ \langle Q5 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \star\ \mathring{\mathbf{U}}\ \langle Q3 \rangle\ )$	$\mid$	$\mathbf{A}(\ \langle Q5 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{W}}\ \langle Q3 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \mathring{\mathbf{W}}\ \langle Q4 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{W}}\ \langle Q5 \rangle\ )$	$\mid$	$\star \wedge \langle Q2 \rangle$	$\mid$
		$\mathbf{AX}\ \langle Q2 \rangle$	$\mid$	$\mathbf{A}(\ \langle Q2 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q2 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	;
$\langle Q3 \rangle$	$::=$	$\mathbf{AF}\ \langle Q6 \rangle$	$\mid$	$\mathbf{A}(\ \langle Q1 \rangle\ \mathbf{U}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q2 \rangle\ \mathbf{U}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \langle Q4 \rangle\ \mathbf{U}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q5 \rangle\ \mathbf{U}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q6 \rangle\ \mathbf{U}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \langle Q7 \rangle\ \mathbf{U}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{U}\ \langle Q6 \rangle\ )$	$\mid$	$\star \vee \langle Q3 \rangle$	$\mid$
		$\mathbf{AF}\ \langle Q3 \rangle$	$\mid$	$\mathbf{A}(\ \langle Q3 \rangle\ \mathbf{U}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{U}\ \langle Q3 \rangle\ )$	;
$\langle Q4 \rangle$	$::=$	$\star \vee \langle Q5 \rangle$	$\mid$	$\mathbf{AF}\ \langle Q5 \rangle$	$\mid$	$\mathbf{AF}\ \langle Q7 \rangle$	$\mid$
		$\mathbf{A}(\ \langle Q6 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q7 \rangle\ \mathring{\mathbf{U}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{U}\ \langle Q5 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \mathbf{U}\ \langle Q7 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{U}}\ \langle Q7 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q6 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \bar{\mathbf{U}}\ \langle Q7 \rangle\ )$	$\mid$	$\mathbf{A}(\ \langle Q1 \rangle\ \mathbf{W}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q2 \rangle\ \mathbf{W}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \langle Q3 \rangle\ \mathbf{W}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q5 \rangle\ \mathbf{W}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q6 \rangle\ \mathbf{W}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \langle Q7 \rangle\ \mathbf{W}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q6 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	$\mid$	$\mathbf{A}(\ \langle Q7 \rangle\ \mathring{\mathbf{W}}\ \star\ )$	$\mid$
		$\mathbf{A}(\ \star\ \mathbf{W}\ \langle Q3 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{W}\ \langle Q5 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{W}\ \langle Q6 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \star\ \mathbf{W}\ \langle Q7 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q6 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \bar{\mathbf{W}}\ \langle Q7 \rangle\ )$	$\mid$
		$\star \vee \langle Q4 \rangle$	$\mid$	$\mathbf{AF}\ \langle Q4 \rangle$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{U}\ \langle Q4 \rangle\ )$	$\mid$
		$\mathbf{A}(\ \langle Q4 \rangle\ \mathbf{W}\ \star\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathbf{W}\ \langle Q4 \rangle\ )$	;		
$\langle Q5 \rangle$	$::=$	$\mathbf{A}(\ \star\ \mathring{\mathbf{U}}\ \langle Q6 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{W}}\ \langle Q6 \rangle\ )$	$\mid$	$\mathbf{A}(\ \star\ \mathring{\mathbf{W}}\ \langle Q7 \rangle\ )$	;

Table 4.3: CTLQ^x production rules (1)

$\langle Q6 \rangle$	::=	$\mathbf{A}(\langle Q8 \rangle\ \mathbf{U}\ \star)$	$\mathbf{A}(\langle Q9 \rangle\ \mathbf{U}\ \star)$	$\star \vee \langle Q6 \rangle$	;
$\langle Q7 \rangle$	::=	$\star \wedge \langle Q6 \rangle$	$\star \vee \langle Q8 \rangle$	$\star \vee \langle Q9 \rangle$	|
		$\mathbf{A}(\langle Q8 \rangle\ \mathbf{W}\ \star)$	$\mathbf{A}(\langle Q9 \rangle\ \mathbf{W}\ \star)$	$\star \wedge \langle Q7 \rangle$	|
		$\star \vee \langle Q7 \rangle$	;		
$\langle Q8 \rangle$	::=	$\mathbf{AF}\ \langle Q9 \rangle$	$\mathbf{AG}\ \langle Q1 \rangle$	$\mathbf{AG}\ \langle Q3 \rangle$	|
		$\mathbf{AG}\ \langle Q4 \rangle$	$\mathbf{AG}\ \langle Q6 \rangle$	$\mathbf{AG}\ \langle Q7 \rangle$	|
		$\mathbf{A}(\star\ \mathbf{U}\ \langle Q9 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{U}}\ \langle Q9 \rangle)$	$\mathbf{A}(\star\ \bar{\mathbf{U}}\ \langle Q9 \rangle)$	|
		$\mathbf{A}(\star\ \mathbf{W}\ \langle Q9 \rangle)$	$\mathbf{A}(\star\ \bar{\mathbf{W}}\ \langle Q9 \rangle)$	$\star \wedge \langle Q8 \rangle$	|
		$\mathbf{AX}\ \langle Q8 \rangle$	$\mathbf{AF}\ \langle Q8 \rangle$	$\mathbf{AG}\ \langle Q8 \rangle$	|
		$\mathbf{A}(\langle Q8 \rangle\ \mathring{\mathbf{U}}\ \star)$	$\mathbf{A}(\star\ \mathbf{U}\ \langle Q8 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{U}}\ \langle Q8 \rangle)$	|
		$\mathbf{A}(\star\ \bar{\mathbf{U}}\ \langle Q8 \rangle)$	$\mathbf{A}(\langle Q8 \rangle\ \mathring{\mathbf{W}}\ \star)$	$\mathbf{A}(\star\ \mathbf{W}\ \langle Q8 \rangle)$	|
		$\mathbf{A}(\star\ \bar{\mathbf{W}}\ \langle Q8 \rangle)$	;		
$\langle Q9 \rangle$	::=	$\mathbf{A}(\star\ \mathring{\mathbf{W}}\ \langle Q8 \rangle)$	$\star \wedge \langle Q9 \rangle$	$\mathbf{AX}\ \langle Q9 \rangle$	|
		$\mathbf{A}(\langle Q9 \rangle\ \mathring{\mathbf{U}}\ \star)$	$\mathbf{A}(\langle Q9 \rangle\ \mathring{\mathbf{W}}\ \star)$	$\mathbf{A}(\star\ \mathring{\mathbf{W}}\ \langle Q9 \rangle)$	;
$\langle Q10 \rangle$	::=	$\mathbf{AG}\ \langle Q2 \rangle$	$\mathbf{AG}\ \langle Q5 \rangle$	$\mathbf{AG}\ \langle Q9 \rangle$	|
		$\star \wedge \langle Q10 \rangle$	$\star \vee \langle Q10 \rangle$	$\mathbf{AX}\ \langle Q10 \rangle$	|
		$\mathbf{AF}\ \langle Q10 \rangle$	$\mathbf{AG}\ \langle Q10 \rangle$	$\mathbf{A}(\langle Q10 \rangle\ \mathbf{U}\ \star)$	|
		$\mathbf{A}(\langle Q10 \rangle\ \mathring{\mathbf{U}}\ \star)$	$\mathbf{A}(\star\ \mathbf{U}\ \langle Q10 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{U}}\ \langle Q10 \rangle)$	|
		$\mathbf{A}(\star\ \bar{\mathbf{U}}\ \langle Q10 \rangle)$	$\mathbf{A}(\langle Q10 \rangle\ \mathbf{W}\ \star)$	$\mathbf{A}(\langle Q10 \rangle\ \mathring{\mathbf{W}}\ \star)$	|
		$\mathbf{A}(\star\ \mathbf{W}\ \langle Q10 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{W}}\ \langle Q10 \rangle)$	$\mathbf{A}(\star\ \bar{\mathbf{W}}\ \langle Q10 \rangle)$	;
$\langle Q11 \rangle$	::=	$\mathbf{AF}\ \langle Q1 \rangle$	$\mathbf{AF}\ \langle Q2 \rangle$	$\mathbf{A}(\star\ \mathbf{U}\ \langle Q1 \rangle)$	|
		$\mathbf{A}(\star\ \mathbf{U}\ \langle Q2 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{U}}\ \langle Q1 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{U}}\ \langle Q2 \rangle)$	|
		$\mathbf{A}(\star\ \mathbf{W}\ \langle Q1 \rangle)$	$\mathbf{A}(\star\ \mathbf{W}\ \langle Q2 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{W}}\ \langle Q1 \rangle)$	|
		$\mathbf{A}(\star\ \mathring{\mathbf{W}}\ \langle Q2 \rangle)$	$\star \wedge \langle Q11 \rangle$	$\star \vee \langle Q11 \rangle$	|
		$\mathbf{AX}\ \langle Q11 \rangle$	$\mathbf{AF}\ \langle Q11 \rangle$	$\mathbf{AG}\ \langle Q11 \rangle$	|
		$\mathbf{A}(\langle Q11 \rangle\ \mathbf{U}\ \star)$	$\mathbf{A}(\langle Q11 \rangle\ \mathring{\mathbf{U}}\ \star)$	$\mathbf{A}(\star\ \mathbf{U}\ \langle Q11 \rangle)$	|
		$\mathbf{A}(\star\ \mathring{\mathbf{U}}\ \langle Q11 \rangle)$	$\mathbf{A}(\star\ \bar{\mathbf{U}}\ \langle Q11 \rangle)$	$\mathbf{A}(\langle Q11 \rangle\ \mathbf{W}\ \star)$	|
		$\mathbf{A}(\langle Q11 \rangle\ \mathring{\mathbf{W}}\ \star)$	$\mathbf{A}(\star\ \mathbf{W}\ \langle Q11 \rangle)$	$\mathbf{A}(\star\ \mathring{\mathbf{W}}\ \langle Q11 \rangle)$	|
		$\mathbf{A}(\star\ \bar{\mathbf{W}}\ \langle Q11 \rangle)$	;		

Table 4.4: CTLQ^x production rules (2)

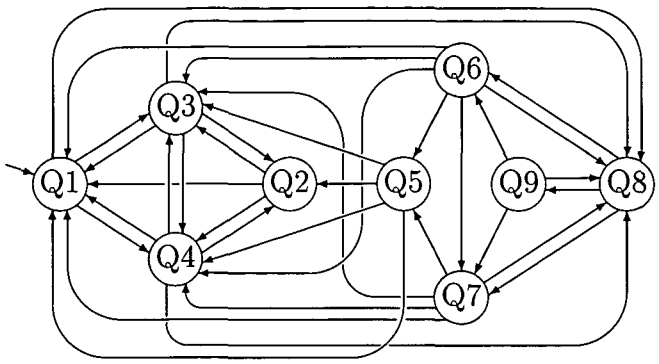


Figure 4.3: CTLQ dependence diagram

The dependencies between the non-terminals in the above grammar are illustrated in Figure 4.3. As in the case for LTL, this graph can be interpreted as an automaton that analyzes a given query starting from the placeholder up to the topmost operator. Its initial state is Q1 because the placeholder occurs in the definition of $\langle Q1 \rangle$. For example, there is a transition from state Q1 to state Q8 because in the definition of nonterminal $\langle Q8 \rangle$ there appears $\mathbf{AG}\,\langle Q1 \rangle$, i.e., there is an operator that leads from nonterminal $\langle Q1 \rangle$ to nonterminal $\langle Q8 \rangle$. For simplicity, we omitted the transitions from each state to itself and the labels on each transition. Since our grammar is deterministic, each query can be uniquely assigned to a node in the graph. For example, it can be easily verified that the query $\mathbf{AF}\,\mathbf{A}((a \wedge ?)\,\mathbf{U}\,b)$ belongs to node Q3. Note that the states corresponding to the nonterminals $\langle Q10 \rangle$ and $\langle Q11 \rangle$ were omitted for simplicity. These states have less interaction with other states because they have only incoming edges (from Q2, Q5, and Q9 to Q10, and from Q1 and Q2 to Q11) but no outgoing edges.

In the following, we will show that all instantiations of templates in CTLQ^x are exact. This will be done by a series of nested inductive proofs on the sublanguages of CTLQ^x. As we shall see soon, a major complication in the proofs arises from the fact that the dependencies between these sublanguages are circular (cf. Chapter 3). Therefore, the non-trivial dependencies as shown in Figure 4.3 are crucial in our proofs.

Remark 4.15. Although the definition of CTLQ^x is quite complex, it does

not contain all exact CTL queries, i.e., there are also exact queries in $\overline{\text{CTLQ}^x}$. However, the existence of a *simple* grammar for *all* exact queries seems to be improbable because it is hard to decide exactness (cf. Theorem 4.3).

4.4.1 Proof of Exactness

This section is devoted to the proof of one of the main results of this thesis, namely the exactness of CTLQ^x. To this aim, we show that all queries in CTLQ^x are collecting. Then, the exactness of CTLQ^x follows by Corollary 4.1. Let us start with the following relation between states.

Definition 4.14 (Reachability). Let s_1 and s_2 be two states in a Kripke structure. Then, state s_2 is *reachable* from state s_1, in symbols $s_1 \rightsquigarrow s_2$, iff there exists a path $\pi \in \text{paths}(s_1)$ such that $\pi(n) = s_2$ for some $n \in \mathbb{N}$.

Moreover, since the collecting property introduced in Section 4.2 is too weak, we need the following stronger variants.

Definition 4.15 (Collecting properties). Let γ be a CTL query.
We say γ is *strong collecting* iff for all states s_1 and s_2 such that $s_1 \rightsquigarrow s_2$ and all formulas φ and ψ:
If $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, then $s_2 \models \gamma[\varphi \wedge \psi]$.
We say γ is *boundary collecting* iff for all states s_1 and s_2 such that $s_1 \rightsquigarrow s_2$ and all formulas φ and ψ:
If $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, then $s_2 \models \gamma[\varphi \wedge \psi]$ or $s_1 \models \gamma[\bot]$.
We say γ is *intermediate collecting* iff for all states s_1 and s_2 such that $s_1 \rightsquigarrow s_2$ and all formulas φ and ψ:
If $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, then $s_2 \models \gamma[\varphi \wedge \psi]$ or there exists $r < n$ such that $\pi^r \models \gamma[\bot]$ for all paths $\pi \in \text{paths}(s_1)$ with $\pi(n) = s_2$ for some $n \in \mathbb{N}$.
We say γ is *weak collecting* iff it is collecting.

Note that every strong collecting query is also boundary collecting, every boundary collecting query is also intermediate collecting (consider the case $r = 0$ together with Lemma 2.1), and every intermediate collecting query is also weak collecting (consider the case $s_1 = s_2$).

Now, let us start to prove the exactness of CTLQ^x. This will be done by a series of auxiliary results using the above collecting properties.

The following lemma is an auxiliary result on CTLQ^6 and CTLQ^7. Since $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$ and $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$ depend on each other (cf. Figure 4.3), we have to make preliminary assumptions on subqueries in CTLQ^8 and CTLQ^9.

Lemma 4.23. *Let $\gamma \in CTLQ^6 \cup CTLQ^7$. Suppose that every subquery in $CTLQ^8$ and $CTLQ^9$ is strong collecting. Then, γ is boundary collecting.*

Proof. Structural induction on γ. See Appendix B.1 for details. □

The following lemma is an auxiliary result on CTLQ^1 and CTLQ^2. Since $\mathrm{CTLQ}^1 \cup \mathrm{CTLQ}^2$ and $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$ depend on each other (cf. Figure 4.3), we have to make preliminary assumptions on subqueries in CTLQ^3, CTLQ^4, and CTLQ^5. In addition, to be able to use Lemma 4.23, we also have to make assumptions on queries in CTLQ^8 and CTLQ^9.

Lemma 4.24. *Let $\gamma \in CTLQ^1 \cup CTLQ^2$. Suppose that every subquery in $CTLQ^3$, $CTLQ^4$, and $CTLQ^5$ is intermediate collecting, and every subquery in $CTLQ^8$ and $CTLQ^9$ is strong collecting. Then, γ is weak collecting.*

Proof. Structural induction on γ. See Appendix B.1 for details. □

The following lemma is an auxiliary result on CTLQ^3, CTLQ^4 and CTLQ^5. Since $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$ and $\mathrm{CTLQ}^1 \cup \mathrm{CTLQ}^2$ depend on each other (cf. Figure 4.3), we have to make preliminary assumptions on subqueries in CTLQ^1 and CTLQ^2. In addition, to be able to use Lemma 4.23, we also have to make assumptions on queries in CTLQ^8 and CTLQ^9.

Lemma 4.25. *Let $\gamma \in CTLQ^3 \cup CTLQ^4 \cup CTLQ^5$. Suppose that every subquery in $CTLQ^1$ and $CTLQ^2$ is weak collecting, and every subquery in $CTLQ^8$ and $CTLQ^9$ is strong collecting. Then, γ is intermediate collecting.*

Proof. Structural induction on γ. See Appendix B.1 for details. □

The following lemma is an auxiliary result on CTLQ^8 and CTLQ^9. Since $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$ and $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$ depend on each other (cf. Figure 4.3), we have to make preliminary assumptions on subqueries in CTLQ^6 and CTLQ^7. In addition, we need assumptions on queries in CTLQ^1, CTLQ^3, and CTLQ^4.

Lemma 4.26. *Let $\gamma \in CTLQ^8 \cup CTLQ^9$. Suppose that every subquery in $CTLQ^1$, $CTLQ^3$, $CTLQ^4$, $CTLQ^6$, and $CTLQ^7$ is weak collecting. Then, γ is strong collecting.*

Proof. Structural induction on γ. See Appendix B.1 for details. □

Now, we have finished our basic results. In the following, we will successively reduce the number of assumptions in the above lemmas; in particular, we will first remove the assumptions on CTLQ^6 and CTLQ^7 subqueries. In order to obtain the assertion of Lemma 4.26 without its assumptions on CTLQ^6 and CTLQ^7 subqueries, we use an inductive proof on the number of subqueries in $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$.

Lemma 4.27. *Let $\gamma \in CTLQ^8 \cup CTLQ^9$. Suppose that every subquery in $CTLQ^1$, $CTLQ^3$, and $CTLQ^4$ is weak collecting. Then, γ is strong collecting.*

Proof. Induction on the number of subqueries in $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$.
Induction start: If γ contains no subquery in $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$, then the assumptions of Lemma 4.26 on subqueries in CTLQ^6 and CTLQ^7 are trivially satisfied. Thus, the only remaining assumptions of Lemma 4.26 are on queries in CTLQ^1, CTLQ^3, and CTLQ^4, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.26 to γ and obtain that γ is strong collecting.
Induction step: Let $\bar{\gamma}$ be any CTLQ^6 or CTLQ^7 subquery of γ, and $\bar{\bar{\gamma}}$ be any CTLQ^8 or CTLQ^9 subquery of $\bar{\gamma}$. Note that, by definition, every CTLQ^6 and CTLQ^7 query has a CTLQ^8 or CTLQ^9 subquery (cf. Figure 4.3). Since the number of $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$ subqueries of $\bar{\bar{\gamma}}$ must be less than the number of $\mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$ subqueries of γ, we can apply the induction hypothesis and obtain that $\bar{\bar{\gamma}}$ is strong collecting. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumptions of Lemma 4.23 are satisfied. So we can apply Lemma 4.23 to $\bar{\gamma}$ and obtain that $\bar{\gamma}$ is boundary collecting. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumptions of Lemma 4.26 on subqueries in CTLQ^6 and CTLQ^7 are satisfied. Thus, the only remaining assumptions of Lemma 4.26 are on queries in CTLQ^1, CTLQ^3, and CTLQ^4, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.26 to γ and obtain that γ is strong collecting. □

In order to obtain the assertion of Lemma 4.24 without its assumptions on CTLQ^8 and CTLQ^9 subqueries, we use an inductive proof on the number of subqueries in $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$.

Lemma 4.28. *Let $\gamma \in CTLQ^1 \cup CTLQ^2$. Suppose that every subquery in $CTLQ^3$, $CTLQ^4$, and $CTLQ^5$ is intermediate collecting. Then, γ is weak collecting.*

Proof. Induction on the number of subqueries in $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$.
Induction start: If γ contains no subquery in $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$, then the assumptions of Lemma 4.24 on subqueries in CTLQ^8 and CTLQ^9 are trivially satisfied. Thus, the only remaining assumptions of Lemma 4.24 are on queries in CTLQ^3, CTLQ^4, and CTLQ^5, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.24 to γ and obtain that γ is weak collecting.
Induction step: Let $\bar{\gamma}$ be any CTLQ^8 or CTLQ^9 subquery of γ, and $\bar{\bar{\gamma}}$ be any CTLQ^1 subquery of $\bar{\gamma}$. Note that, by definition, every CTLQ^8 and CTLQ^9 query has a CTLQ^1 subquery (cf. Figure 4.3). Since the number of $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$ subqueries of $\bar{\bar{\gamma}}$ must be less than the number of $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$ subqueries of γ, we can apply the induction hypothesis and obtain that $\bar{\bar{\gamma}}$ is weak collecting. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumption of Lemma 4.27 on subqueries in CTLQ^1 is satisfied. So the only remaining assumptions of Lemma 4.27 are on queries in CTLQ^3 and CTLQ^4, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.27 to $\bar{\gamma}$ and obtain that $\bar{\gamma}$ is strong collecting. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumptions of Lemma 4.24 on subqueries in CTLQ^8 and CTLQ^9 are satisfied. Thus, the only remaining assumptions of Lemma 4.24 are on queries in CTLQ^3, CTLQ^4, and CTLQ^5, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.24 to γ and obtain that γ is weak collecting. □

In order to obtain the assertion of Lemma 4.25 without its assumptions on CTLQ^8 and CTLQ^9 subqueries, we use an inductive proof on the number of subqueries in $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$.

Lemma 4.29. *Let $\gamma \in CTLQ^3 \cup CTLQ^4 \cup CTLQ^5$. Suppose that every subquery in $CTLQ^1$ and $CTLQ^2$ is weak collecting. Then, γ is intermediate collecting.*

Proof. Induction on the number of subqueries in $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$.
Induction start: If γ contains no subquery in $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$, then the assumptions of Lemma 4.25 on subqueries in CTLQ^8 and CTLQ^9 are trivially satisfied. Thus, the only remaining assumptions of Lemma 4.25 are on queries in CTLQ^1 and CTLQ^2, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.25 to γ and obtain that γ is intermediate collecting.
Induction step: Let $\bar{\gamma}$ be any CTLQ^8 or CTLQ^9 subquery of γ. If $\bar{\gamma}$ contains no subquery in $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4$, then the assumptions of Lemma 4.27 on queries in CTLQ^3 and CTLQ^4 are trivially satisfied. Otherwise, let $\bar{\bar{\gamma}}$ be any CTLQ^3 or CTLQ^4 subquery of $\bar{\gamma}$. Since the number of $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$ subqueries of $\bar{\bar{\gamma}}$ must be less than the number of $\mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$ subqueries of γ, we can apply the induction hypothesis and obtain that $\bar{\bar{\gamma}}$ is intermediate collecting. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumptions of Lemma 4.27 on subqueries in CTLQ^3 and CTLQ^4 are again satisfied. So in both cases the only remaining assumption of Lemma 4.27 is on queries in CTLQ^1, which is satisfied by the actual assumptions. Hence, we can apply Lemma 4.27 to $\bar{\gamma}$ and obtain that $\bar{\gamma}$ is strong collecting. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumptions of Lemma 4.25 on subqueries in CTLQ^8 and CTLQ^9 are satisfied. Thus, the only remaining assumptions of Lemma 4.25 are on queries in CTLQ^1 and CTLQ^2, which are satisfied by the actual assumptions. Hence, we can apply Lemma 4.25 to γ and obtain that γ is intermediate collecting. □

In order to obtain the assertion of Lemma 4.28 without its assumptions, we use an inductive proof on the number of subqueries in $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Thus, we have reached our first result without assumptions.

Lemma 4.30. *Every query in $CTLQ^1$ and $CTLQ^2$ is weak collecting.*

Proof. Induction on the number of subqueries in $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$.
Induction start: If $\gamma \in \mathrm{CTLQ}^1 \cup \mathrm{CTLQ}^2$ contains no subquery in $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$, then the assumptions of Lemma 4.28 are trivially satisfied. Hence, we can apply Lemma 4.28 to γ and obtain that γ is weak collecting.
Induction step: Let $\bar{\gamma}$ be any CTLQ^3, CTLQ^4, or CTLQ^5 subquery of γ, and $\bar{\bar{\gamma}}$ be any CTLQ^1 or CTLQ^2 subquery of $\bar{\gamma}$. Note that, by definition, every CTLQ^3, CTLQ^4, and CTLQ^5 query has at least a CTLQ^1 subquery

(cf. Figure 4.3). Since the number of $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$ subqueries of $\bar{\bar{\gamma}}$ must be less than the number of $\mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$ subqueries of γ, we can apply the induction hypothesis and obtain that $\bar{\bar{\gamma}}$ is weak collecting. Thus, since $\bar{\bar{\gamma}}$ was chosen w.l.o.g., the assumptions of Lemma 4.29 are satisfied. So we can apply Lemma 4.29 to $\bar{\gamma}$ and obtain that $\bar{\gamma}$ is intermediate collecting. Since $\bar{\gamma}$ was chosen w.l.o.g., the assumptions of Lemma 4.28 are satisfied. Hence, we can apply Lemma 4.28 to γ and obtain that γ is weak collecting. □

Since the assumptions of Lemma 4.29 are satisfied according to Lemma 4.30, we trivially obtain the following corollary by Lemma 4.29.

Corollary 4.8. *Every query in $CTLQ^3$, $CTLQ^4$, and $CTLQ^5$ is intermediate collecting.*

Since the assumptions of Lemma 4.27 are satisfied according to Lemma 4.30 and Corollary 4.8, we trivially obtain the following corollary by Lemma 4.27.

Corollary 4.9. *Every query in $CTLQ^8$ and $CTLQ^9$ is strong collecting.*

Since the assumptions of Lemma 4.23 are satisfied according to Corollary 4.9, we trivially obtain the following corollary by Lemma 4.23.

Corollary 4.10. *Every query in $CTLQ^6$ and $CTLQ^7$ is boundary collecting.*

Remark 4.16. Note that, in principle, Lemma 4.30, Corollary 4.8, Corollary 4.9, and Corollary 4.10 can also be obtained from Lemma 4.23, Lemma 4.24, Lemma 4.25, and Lemma 4.26 by successively applying our circular cut rule from Section 3.2.

Now, we have shown the collecting property for almost all sublanguages of CTLQ^x. The only remaining case is CTLQ^{10}. In order to show that CTLQ^{10} is collecting, we need some further auxiliary results.

Lemma 4.31. *Let $\gamma \in CTLQ^3 \cup CTLQ^6$. Then, $\gamma[\top]$ implies $\mathbf{AF}\,\gamma[\bot]$.*

Proof. Structural induction on γ. See Appendix B.1 for details. □

By using Lemma 4.31, it is now possible to show the following property. Intuitively it says that if there globally exists any solution, then everything is globally a solution. This result can then be used to prove the collecting property of CTLQ^{10}; in particular, that CTLQ^{10} is strong collecting.

Lemma 4.32. *Let* $\gamma \in CTLQ^2 \cup CTLQ^5 \cup CTLQ^9$. *Then,* $\mathbf{AG}\,\gamma[\top]$ *implies* $\mathbf{AG}\,\gamma[\bot]$.

Proof. Structural induction on γ. See Appendix B.1 for details. □

Note that every query in CTLQ^{10} contains a subquery of the form $\mathbf{AG}\,\bar{\gamma}$, where $\bar{\gamma} \in \mathrm{CTLQ}^2 \cup \mathrm{CTLQ}^5 \cup \mathrm{CTLQ}^9$. Therefore, we can use Lemma 4.32 as induction start in the proof of the following lemma.

Lemma 4.33. *Let* $\gamma \in CTLQ^{10}$. *Then,* $\gamma[\top]$ *implies* $\gamma[\bot]$.

Proof. Structural induction on γ. See Appendix B.1 for details. □

The following corollary is directly implied by Lemma 4.33.

Corollary 4.11. *Every query in* $CTLQ^{10}$ *is strong collecting.*

Proof. Let $\gamma \in \mathrm{CTLQ}^{10}$ and s_1 as well as s_2 be two states in a Kripke structure such that $s_1 \rightsquigarrow s_2$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$. Then, by Lemma 2.1, we know that $s_2 \models \gamma[\top]$. Thus, by Lemma 4.33, we obtain $s_2 \models \gamma[\bot]$, which implies $s_2 \models \gamma[\varphi \wedge \psi]$ by Lemma 2.1. □

Now, recall that CTLQ^x is defined as the union of the above considered CTL query languages (cf. Definition 4.13). Moreover, as already mentioned, every strong collecting query is boundary collecting, every boundary collecting query is intermediate collecting, and every intermediate collecting query is (weak) collecting. Thus, we obtain the following corollary by Lemma 4.30, Corollary 4.8, Corollary 4.9, Corollary 4.10, and Corollary 4.11.

Corollary 4.12. *Every query in* $CTLQ^x$ *is collecting.*

Hence, since CTLQ^x is a subset of ACTLQ^m, we obtain by Lemma 4.22, Corollary 4.12, and Corollary 4.1 one of the main results of this thesis.

Theorem 4.6. *Every query in* $CTLQ^x$ *is exact.*

In contrast to the characterization of LTLQ^x, we are unfortunately not able to prove the maximality of CTLQ^x in the sense that all simple queries in $\overline{\mathrm{CTLQ}^x}$ are not exact. We will now show that CTLQ^x is in fact *not* maximal in this sense. To this aim, consider the simple query $\gamma = \mathbf{AF}(a \wedge \mathbf{AF}(b \vee \mathbf{AG}\,?))$. It can be easily verified that $\gamma \in \overline{\mathrm{CTLQ}^x}$ although γ is collecting as stated in the following proposition.

Proposition 4.3. *The query* $\gamma = \mathbf{AF}(a \wedge \mathbf{AF}(b \vee \mathbf{AG}\,?))$ *is collecting.*

Proof. Let s be any state in a Kripke structure such that $s \models \gamma[\varphi] \wedge \gamma[\psi]$ for some formulas φ and ψ. Moreover, let $\bar{\gamma} = \mathbf{AF}(b \vee \mathbf{AG}\,?)$, that is, $\gamma = \mathbf{AF}(a \wedge \bar{\gamma})$. Now, w.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Thus, we know that $\pi \models \mathbf{F}(a \wedge \bar{\gamma}[\varphi]) \wedge \mathbf{F}(a \wedge \bar{\gamma}[\psi])$. Hence, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models a \wedge \bar{\gamma}[\varphi]$ or $\pi^k \models a \wedge \bar{\gamma}[\psi]$. W.l.o.g., we can assume that $\pi^k \models a \wedge \bar{\gamma}[\varphi]$. Now, w.l.o.g., we choose any path $\sigma \in \mathrm{paths}(\pi(k))$. Thus, we know that $\sigma \models a \wedge \bar{\gamma}[\varphi]$ and $\sigma^l \models a \wedge \bar{\gamma}[\psi]$ for some $l \in \mathbb{N}$. In particular, this implies $\sigma \models \bar{\gamma}[\varphi]$ and $\sigma^l \models \bar{\gamma}[\psi]$. It can be easily verified that $\bar{\gamma} \in \mathrm{CTLQ}^4$. Thus, by Corollary 4.8, we know that $\bar{\gamma}$ is intermediate collecting. Hence, we obtain $\sigma^l \models \bar{\gamma}[\varphi \wedge \psi]$ or there exists $r < l$ and $\sigma^r \models \bar{\gamma}[\bot]$. By Lemma 2.1, this implies $\sigma \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$. Since $\sigma \in \mathrm{paths}(\pi(k))$ was chosen w.l.o.g., we know that $\pi^k \models \mathbf{AF}\,\bar{\gamma}[\varphi \wedge \psi]$. It is easy to see that this is equivalent to $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi^k \models a$, we obtain $\pi^k \models a \wedge \bar{\gamma}[\varphi \wedge \psi]$, which implies $\pi \models \mathbf{F}(a \wedge \bar{\gamma}[\varphi \wedge \psi])$. Since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we know that $s \models \gamma[\varphi \wedge \psi]$. Hence, γ is collecting. □

Finally, let us remark that a proof of maximality by counterexample construction as in the case of LTLQ^x in Section 4.3.2 is much more difficult in the case of CTLQ^x. In particular, a counterexample to the collecting property for queries in CTLQ^x is in general a computation tree instead of a computation path as in the case of LTLQ^x.

4.5 Summary

An *exact solution* to a temporal logic query γ in a Kripke structure $\mathfrak{K}$ is a temporal logic formula ξ such that $\mathfrak{K} \models \gamma[\xi]$ and $\xi \Rightarrow \varphi$ iff $\mathfrak{K} \models \gamma[\varphi]$, i.e., a solution that exactly characterizes the set of all solutions. A temporal logic query is *exact* iff it has an exact solution in every Kripke structure where the set of solutions is not empty.

Exact temporal logic queries can be semantically characterized in several equivalent ways. The most important one in this chapter is the following: A temporal logic query is exact iff it is *monotonic* and *collecting*. Monotonic in this context means that $\varphi \Rightarrow \psi$ implies $\gamma[\varphi] \Rightarrow \gamma[\psi]$, and collecting means that $\gamma[\varphi] \wedge \gamma[\psi]$ implies $\gamma[\varphi \wedge \psi]$ for all formulas φ and ψ.

Such a characterization of exact temporal logic queries can be used to obtain syntactic fragments of exact LTL and CTL queries. In particular, Section 4.3 presents a syntactic fragment LTLQ^x of exact LTL queries. A proof that all queries in this fragment are indeed exact is given in Section 4.3.1, and a proof that this fragment is maximal in the sense of a template characterization is given in Section 4.3.2. In addition, Section 4.4 presents a syntactic fragment CTLQ^x of exact CTL queries. A proof that all queries in this fragment are indeed exact is given in Section 4.4.1. In contrast to LTLQ^x, the maximality of CTLQ^x could not be shown.

Exactness of temporal logic queries is an interesting property on its own because it may give a better understanding of temporal logics. In particular, the key property in this context is *distributivity*, another characterization of exact temporal logic queries. A temporal logic query γ is distributive (over conjunction) iff γ satisfies $\gamma[\varphi] \wedge \gamma[\psi] \Leftrightarrow \gamma[\varphi \wedge \psi]$ for all formulas φ and ψ. Each such distributive temporal logic query amounts to a set of equivalences between temporal logic formulas.

Moreover, the auxiliary results used in our exactness proofs of temporal logic queries show another interesting property. In particular, these results enable us to eliminate existential choices when evaluating temporal operators. This can be exploited in order to obtain efficient symbolic query solving algorithms for queries in CTLQ^x as discussed in Chapter 5.

Chapter 5

Solving Temporal Logic Queries

5.1 Introduction

A natural question when investigating temporal logic queries is about algorithms to solve them. There exist several such algorithms in the literature which we will describe at the end of this chapter. Our focus, however, lies on the symbolic query solving algorithm introduced by Chan [Cha00]. Although other algorithms are more general, a thorough investigation of Chan's algorithm is interesting for several reasons.

Chan presented this algorithm in order to solve queries in his syntactic fragment of *valid* CTL queries, but he neither proved its correctness nor did he describe the underlying intuition and mathematical principles. Thus, it was unknown how and why the algorithm works. Moreover, its correctness was uncertain in consideration of the fact that Chan's fragment of valid CTL queries was shown to be erroneous [Sam02, SV03].

We will prove that the Chan algorithm is correct for all queries in CTLQ^x that have a solution in every Kripke structure. The collecting properties introduced in the previous chapter are crucial in this proof since they enable us to eliminate some kind of non-determinism that appears in the evaluation of temporal operators as shown in the following example.

Example 5.1. Consider the LTL formula $\varphi \mathbf{U} \psi$, which we want to check on path π. To this aim, let n be the least number such that $\pi^n \not\models \varphi$. Trivially, this implies $\pi^i \models \varphi$ for all $i < n$. Now, by the semantics of the strong until operator, we know that $\pi \models \varphi \mathbf{U} \psi$ iff there exists $i \leq n$ such that $\pi^i \models \psi$. These existential choices are the kind of non-determinism we

are talking about in this chapter, i.e., in the worst case we have to check ψ at all positions $i \leq n$ although only a single witness is needed.

The key observation exploited in Chan's algorithm is now that if ψ is of a special form, then it suffices to check ψ at a single position on π, i.e., the strong until operator can be determinized. In our terminology, if $\psi = \gamma[\psi']$ such that γ is intermediate collecting for a subformula ψ', then $\pi \models \varphi \mathbf{U} \psi$ iff $\pi^{i_0} \models \psi$, where $i_0 \leq n$ is the highest number such that $\pi^{i_0} \models \gamma[\top]$.

The correctness of the Chan algorithm is important on its own since it is the only symbolic query solving algorithm known so far. Moreover, it reveals the connection to the properties of exact CTL queries proved in the previous chapter and justifies the corresponding exactness proofs. Finally, a thorough understanding of Chan's original algorithm points us the way to an extension in order to solve queries in the whole fragment CTLQ^x. The definition of such an extension and the corresponding correctness proof is the most important part of this chapter. Independently of these results, we will also show how other query solving algorithms can be extended in order to compute non-propositional solutions as well.

This chapter is organized as follows: In Section 5.2, we investigate CTLQ^x query solving based on the Chan algorithm. In particular, Section 5.2.1 shows how the collecting properties can be used to eliminate non-determinism. Based on these results, we prove the correctness of the Chan algorithm in Section 5.2.2. Afterwards, in Section 5.2.3, we present an extension of the Chan algorithm for solving queries in the whole fragment CTLQ^x and prove its correctness. The computation of non-propositional solutions is then separately considered in Section 5.2.4. In Section 5.3, we summarize further query solving algorithms and indicate how they can be extended in order to compute non-propositional solutions. Finally, we summarize in Section 5.4.

5.2 Solving Queries in CTLQ^x

In this section, we show how the theoretical results of the previous chapter can be used for eliminating non-determinism in order to obtain efficient query solving algorithms. In particular, we investigate non-determinism that appears when evaluating temporal operators, how it can be eliminated

by exploiting the collecting properties, and study Chan's symbolic algorithm. A thorough understanding of this algorithm will point us the way to an extension in order to solve queries in the whole fragment CTLQ^x. The most important part in this section is the definition of such an extended algorithm and the proof of its correctness. As already mentioned, the collecting properties proved in the previous section are crucial in the correctness proofs of both the Chan and the extended Chan algorithm.

In order to get an idea of the relationship between query solving and the collecting properties, note that evaluating a temporal logic formula or solving a temporal logic query comprises some kind of non-determinism in general. For example, the existential path quantifier enables us to express non-determinism on paths because it claims that there is a path satisfying some property, but we do not know which one. Similarly, the future operator enables us to express non-determinism on states. Therefore, when checking $s \models \mathbf{E}\varphi$ in a naive way, we have to check $\pi \models \varphi$ in the worst case for all $\pi \in \mathrm{paths}(s)$. Similarly, when checking $\pi \models \mathbf{F}\varphi$, we have to check $\pi^n \models \varphi$ in the worst case for all $n \in \mathbb{N}$.

Obviously, the same holds when solving a query, i.e., in order to obtain all solutions in the presence of non-determinism, a query must in general be solved for all non-deterministic choices. This is easy to see because if one case is omitted, this case could have provided a new solution that is not subsumed by the ones computed so far. Moreover, even if we know that there exists an exact solution, all non-deterministic choices must be checked since otherwise we would not know whether the strongest solution found so far is the strongest solution among all solutions.

The connection between computing an exact solution to queries in CTLQ^x and the collecting properties proved in the previous chapter is: The collecting properties enable us to eliminate the above kinds of non-determinism. In order to locate the cause of non-determinism in this context more systematically, note that temporal operators can be divided into *universal* and *existential* ones. Let us consider some examples.

Example 5.2. For every path π and formula φ it holds that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ if and only if $\forall i \in \mathbb{N}.\ \pi^i \models \bar{\gamma}[\varphi]$. Thus, solving a query $\mathbf{G}\,\bar{\gamma}$ can be reduced to solving its subquery $\bar{\gamma}$ at universally quantified positions on a path. Hence, we classify the global operator $\mathbf{G}$ to be a universal operator.

In contrast, consider path π in Figure 5.1. Obviously, it holds that

$\pi \models a\,\mathbf{U}\,\bar{\gamma}[\varphi]$, but $a\,\mathbf{U}\,\bar{\gamma}$ cannot be reduced to solving its subquery $\bar{\gamma}$ at universally quantified positions on π. However, for every path π it holds that $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$ if and only if $\exists i \in \{j \in \mathbb{N} \mid j \leq n\}.\ \pi^i \models \bar{\gamma}[\varphi]$, where $n \in \mathbb{N}$ is the least number such that $\pi^n \not\models \theta$. Thus, solving a query $\theta\,\mathbf{U}\,\bar{\gamma}$ can be reduced to solving its subquery $\bar{\gamma}$ at existentially quantified positions on a path. Hence, we classify the strong until operator $\mathbf{U}$ with respect to its second argument to be an existential operator.

Note, however, that the strong until operator with respect to its first argument is universal. To see this, let $n \in \mathbb{N}$ be the least number such that $\pi^n \models \theta$ for any path π and formula θ. Then, for every formula φ, it holds that $\pi \models \bar{\gamma}[\varphi]\,\mathbf{U}\,\theta$ if and only if $\forall i \in \{j \in \mathbb{N} \mid j < n\}.\ \pi^i \models \bar{\gamma}[\varphi]$.

Consequently, the kind of non-determinism we consider in this section arises from existential choices when solving a query top-down by a reduction to solving its subqueries. The formal starting point of our investigations is therefore the following definition.

Definition 5.1 (Universal, Existential). Let $\mathbf{O}$ be a temporal operator. Then, we define $\mathbf{O}$ to be *universal* resp. *existential* with respect to its operand ψ on a path π iff for all queries $\gamma = \gamma_{\mathbf{O}}[\bar{\gamma}]$ such that $\gamma_{\mathbf{O}}$ consists only of operator $\mathbf{O}$ and the placeholder occurs only at the position of operand ψ in $\gamma_{\mathbf{O}}$, there exists a set $\mathcal{I} \subseteq \mathbb{N}$ such that for all formulas φ

$$\pi \models \gamma[\varphi] \text{ iff } \begin{cases} \forall i \in \mathcal{I}.\ \pi^i \models \bar{\gamma}[\varphi] & : \textit{universal} \\ \exists i \in \mathcal{I}.\ \pi^i \models \bar{\gamma}[\varphi] & : \textit{existential} \end{cases}$$

We define $\mathbf{O}$ to be *universal* (resp. *existential*) with respect to its operand ψ iff it is universal (resp. not universal, but existential) on every path π satisfying $\pi \models \gamma_{\mathbf{O}}[\top]$ (resp. $\pi \not\models \gamma_{\mathbf{O}}[\bot]$) for every query $\gamma_{\mathbf{O}}$ defined as above. Accordingly, we call $\bar{\gamma}$ *universally occurring* resp. *existentially occurring* in every query that contains such a query γ above as subquery.

Note that all temporal operators used in this thesis are either universal or existential with respect to a selected operand. This fact is summarized in Table 5.1, where the placeholder indicates the corresponding operand. It can be easily verified that this classification is consistent with the above definition. For example, let $\gamma = \mathbf{AG}(a \vee \mathbf{A}(b\,\mathbf{U}\,\mathbf{AX}\,\bar{\gamma}))$. Then, $\mathbf{AX}\,\bar{\gamma}$ is existentially occurring in γ, whereas $\bar{\gamma}$ is universally occurring.

Universal				Existential			
$\mathbf{X}?$	$?\mathbf{U}\varphi$	$?\mathring{\mathbf{U}}\varphi$	$\varphi\bar{\mathbf{U}}?$	$\mathbf{F}?$			
$\mathbf{G}?$	$?\mathbf{W}\varphi$	$?\mathring{\mathbf{W}}\varphi$	$\varphi\bar{\mathbf{W}}?$	$\varphi\mathbf{U}?$	$\varphi\mathring{\mathbf{U}}?$	$\varphi\mathbf{W}?$	$\varphi\mathring{\mathbf{W}}?$

Table 5.1: Classification of temporal operators

Note that also the path quantifiers can be classified in this way. In particular, the universal path quantifier is universal and the existential path quantifier is existential. However, since the existential path quantifier is not allowed in CTLQ^x, we do not need to take care of this.

The following definition will enable us to easily describe an appropriate set $\mathcal{I}$ according to Definition 5.1 for the existential operators in Table 5.1.

Definition 5.2 (Prefix indices). A set $\mathcal{I} \subseteq \mathbb{N}$ of natural numbers is a set of *prefix indices* iff for each $n \in \mathcal{I}$ it holds that for all $i < n$, $i \in \mathcal{I}$. In particular, for any path π and formula φ, we define the set of prefix indices

$$\mathcal{I}_\pi(\varphi) = \{n \in \mathbb{N} \mid \forall i < n.\ \pi^i \models \varphi\}.$$

Note that a set of prefix indices is either an initial segment of $\mathbb{N}$ (i.e., a set of the form $\{i \in \mathbb{N} \mid i \leq n\}$ for some $n \in \mathbb{N}$) or $\mathbb{N}$ itself. For example, let π be the path shown in Figure 5.1. Then, $\mathcal{I}_\pi(a) = \{0, 1, 2\}$, $\mathcal{I}_\pi(b) = \{0\}$, and $\mathcal{I}_\pi(a \vee b) = \mathbb{N}$. The following lemma shows that for the strong until operator with respect to its second argument, the set of prefix indices is an appropriate set $\mathcal{I}$ according to Definition 5.1. It follows immediately from the definition of the strong until operator and the set of prefix indices.

Lemma 5.1. *For every path π, query γ, and formulas θ and φ it holds that $\pi \models \theta\,\mathbf{U}\,\gamma[\varphi]$ iff $\exists i \in \mathcal{I}_\pi(\theta).\ \pi^i \models \gamma[\varphi]$.*

Note that all existential operators in Table 5.1 are variants of the strong until operator. Thus, by Lemma 5.1, we obtain also for the other existential operators a set of indices encoding the existential choices.

Remark 5.1. Not surprisingly, there is a close relationship between the classification into universal and existential temporal operators and the first-order quantification of the corresponding operands in the definition of their

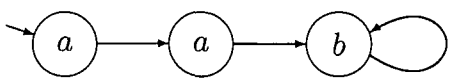


Figure 5.1: Prefix index example

semantics (cf. Table 2.4). For example, the semantics of $\pi \models \varphi \,\mathbf{U}\, \psi$ is defined by $\exists n\ \forall i < n.\ \pi^i \models \varphi$ and $\pi^n \models \psi$. Thus, the first operand φ of the strong until operator is under universal quantification and its second operand ψ is under existential quantification. Accordingly, the strong until operator is universal with respect to its first operand and existential with respect to its second operand. However, there are also universal operators whose corresponding operand is existentially quantified, e.g., the disjoint strong until operator $\bar{\mathbf{U}}$ with respect to its second operand. But in these cases it is easy to see that the semantics of the operators can be equivalently redefined by using a uniqueness quantifier instead of the existential quantifier, which justifies our classification.

5.2.1 Eliminating Non-determinism

The following fact enables us to build up on the results of the previous chapter in order to eliminate non-determinism in the form of existential choices when solving queries in CTLQ^x. To this aim, recall that a query γ is *intermediate collecting* iff for all paths π and formulas φ and ψ it holds that: If $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$, then $\pi^n \models \gamma[\varphi \wedge \psi]$ or there exists $r < n$ such that $\pi^r \models \gamma[\bot]$.[1]

Proposition 5.1. *All existentially occurring subqueries of queries in our fragment CTLQ^x are intermediate collecting.*

Proof. By Table 4.3 and Table 4.4, it is easy to see that all existentially occurring subqueries (i.e., immediate subqueries of existential operators according to Table 5.1) of queries in CTLQ^x are in CTLQ^3, CTLQ^4, CTLQ^5, CTLQ^6, CTLQ^7, CTLQ^8, CTLQ^9, and CTLQ^{10} only. Hence, by Corollary 4.8, Corollary 4.9, Corollary 4.10, and Corollary 4.11, we know that these queries are (at least) intermediate collecting. □

[1] Note that this was our definition of *intermediate collecting* for LTL queries. It is easy to see that it can also be used in the case of CTL queries.

Now, we show in the following lemma that these intermediate collecting queries have a nice property on which our further results are based.

Lemma 5.2. *Let γ be an intermediate collecting query and π be a path. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\top]$ for some $n \in \mathbb{N}$. If for all $i < n$ it holds that $\pi^i \not\models \gamma[\bot]$, then $\pi^n \models \gamma[\varphi]$.*

Proof. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\top]$. Then, by definition of the intermediate collecting property, we obtain $\pi^n \models \gamma[\varphi \wedge \top]$ or there exists $r < n$ such that $\pi^r \models \gamma[\bot]$. Hence, if for all $i < n$ it holds that $\pi^i \not\models \gamma[\bot]$, we know that $\pi^n \models \gamma[\varphi]$. □

Since evaluating an existentially occurring query has in general to be done at several states on a path, we need the following definition.

Definition 5.3 (Solutions). Let γ be a query, π be a path, and $\mathcal{I} \subseteq \mathbb{N}$. Then, we define the set of solutions to γ at positions in $\mathcal{I}$ on π by

$$\mathrm{sol}_{\mathcal{I}}(\pi, \gamma) = \bigcup_{i \in \mathcal{I}} \{\varphi \mid \pi^i \models \gamma[\varphi]\}.$$

For our purposes, the set $\mathcal{I}$ will be a set of prefix indices representing the existential choices of positions on π when solving an existentially occurring subquery γ. Thus, in order to obtain *all* solutions, γ has in general to be solved at all states with indices in $\mathcal{I}$. Note that if there exists $i \in \mathcal{I}$ such that $\pi^i \models \gamma[\bot]$, then by Lemma 2.1 *every* formula is an element of $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$ and therefore the solutions at other states with indices in $\mathcal{I}$ do not affect $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$. This case is somehow exceptional since it can be simply checked by evaluating the formula $\gamma[\bot]$ at all states with indices in $\mathcal{I}$, which can be performed by a single *symbolic* model checking call.

Otherwise, if no such a prefix index exists, it follows immediately from the intermediate collecting property according to Lemma 5.2 that

$$\mathrm{sol}_{\{i_0\}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{i_1\}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{i_2\}}(\pi, \gamma) \subseteq \cdots,$$

where $i_0 < i_1 < i_2 < \cdots$ and $\pi^n \models \gamma[\top]$ for all $n \in \{i_0, i_1, i_2, \ldots\} \subseteq \mathcal{I}$. Hence, when solving an intermediate collecting query at several states on a path, it suffices to solve the query at states with indices as high as possible. In particular, if there exists a highest index $n \in \mathcal{I}$ such that $\pi^n \models \gamma[\top]$, it suffices to solve the query at this single state.

Example 5.3. Consider path π shown in Figure 5.1 and query $\gamma = a\,\mathbf{U}\,\bar{\gamma}$. Now, we want to solve γ on π by reducing it to solving $\bar{\gamma}$ on π. Since the strong until operator with respect to its second argument is an existential operator (cf. Table 5.1), i.e., $\bar{\gamma}$ is an existentially occurring subquery, we know that there exists a reduction of the form $\pi \models \gamma[\varphi]$ iff $\exists i \in \mathcal{I}.\ \pi^i \models \bar{\gamma}[\varphi]$ for some set $\mathcal{I} \subseteq \mathbb{N}$. By Lemma 5.1, we are allowed to choose $\mathcal{I} = \mathcal{I}_\pi(a) = \{0, 1, 2\}$. Hence, we know that solving γ on π can be reduced to solving $\bar{\gamma}$ at states with indices in $\mathcal{I}_\pi(a)$ on π, i.e., we obtain all solutions to γ on π by computing the solutions to $\bar{\gamma}$ on π^0, π^1, and π^2. In this simple example, it is easy to see that this reduction is correct.

Let us now consider the case where $\bar{\gamma}$ is intermediate collecting. In this case we can determinize the above reduction in such a way that $\mathcal{I}$ becomes a singleton set. In particular, if $\pi^i \models \bar{\gamma}[\top]$ and $\pi^i \not\models \bar{\gamma}[\bot]$ for all $i \in \mathcal{I}_\pi(a)$, it follows from the intermediate collecting property according to Lemma 5.2 that $\mathrm{sol}_{\{0\}}(\pi, \bar{\gamma}) \subseteq \mathrm{sol}_{\{1\}}(\pi, \bar{\gamma}) \subseteq \mathrm{sol}_{\{2\}}(\pi, \bar{\gamma})$. Thus, the solutions obtained by solving $\bar{\gamma}$ on π^0 and π^1 are also solutions on π^2. Hence, it suffices to solve $\bar{\gamma}$ on π^2, i.e., the state with highest index in $\mathcal{I}_\pi(a)$. So we know that $\pi \models \gamma[\varphi]$ iff $\pi^2 \models \bar{\gamma}[\varphi]$ for all formulas φ.

Since all existentially occurring subqueries of queries in CTLQ^x are intermediate collecting according to Proposition 5.1, this principle of eliminating existential choices can always be applied when solving queries in CTLQ^x. We are now going to prove this insight formally which will enable us to prove the correctness of the Chan algorithm.

Lemma 5.3. *Let γ be an intermediate collecting query, π be a path, and $\mathcal{I} \subseteq \mathbb{N}$. Suppose that there is a least index $m \in \mathcal{I}$ and a highest index $n \in \mathcal{I}$ such that $\pi^m \models \gamma[\top]$ and $\pi^n \models \gamma[\top]$. If $\pi^i \not\models \gamma[\bot]$ for all $m \leq i < n$, then* $\mathrm{sol}_{\{n\}}(\pi, \gamma) = \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$.

Proof. Since $n \in \mathcal{I}$, we know that $\mathrm{sol}_{\{n\}}(\pi, \gamma) \subseteq \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$. In order to show that $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{n\}}(\pi, \gamma)$, let $\varphi \in \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$. Since $\varphi \in \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$, we know that there exists $j \in \mathcal{I}$ such that $\pi^j \models \gamma[\varphi]$. By Lemma 2.1, this implies $\pi^j \models \gamma[\top]$. Since m is the least index such that $\pi^m \models \gamma[\top]$ and n is the highest index such that $\pi^n \models \gamma[\top]$, we know that $m \leq j \leq n$. So we have $\pi^j \models \gamma[\varphi]$ and $\pi^n \models \gamma[\top]$, where $j \leq n$. Thus, if for all $j \leq i < n$ it holds that $\pi^i \not\models \gamma[\bot]$, we obtain by Lemma 5.2 that $\pi^n \models \gamma[\varphi]$, that is, $\varphi \in \mathrm{sol}_{\{n\}}(\pi, \gamma)$. Hence, $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{n\}}(\pi, \gamma)$. □

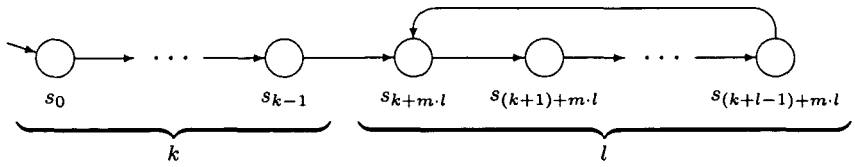


Figure 5.2: Path structure

Obviously, this lemma enables us to eliminate non-determinism, since instead of solving γ at all existential choices encoded in $\mathcal{I}$, it suffices to solve γ at position n. Note, however, that Lemma 5.3 does not cover all cases that may appear when solving a query. In particular, it says nothing about the case where the highest index $n \in \mathcal{I}$ does not exist. This case occurs if $\mathcal{I}$ contains an infinite number of indices i satisfying $\pi^i \models \gamma[\top]$.

Note that such indices must refer to states in a cycle, since only states in a cycle have an infinite number of indices. This can be seen in Figure 5.2, where the general structure of a path is illustrated. The infinite number of indices of each state in a cycle is obtained by varying $m \in \mathbb{N}$.

Definition 5.4 (Cycle indices). For every path π we define the set of *cycle indices* $\text{cycle}(\pi) = \{i \in \mathbb{N} \mid i \geq k\}$, where k is the length of the path prefix of π according to Figure 5.2.

Now, we are able to show the following lemma. Intuitively, it means that the solutions to an intermediate collecting query at states within a cycle are the same at all these states.

Lemma 5.4. *Let γ be an intermediate collecting query and π be a path. Further, let $r, s \in \text{cycle}(\pi)$ such that $\pi^r \models \gamma[\top]$ and $\pi^s \models \gamma[\top]$. If $\pi^i \not\models \gamma[\bot]$ for all $i \in \text{cycle}(\pi)$, then $\text{sol}_{\{r\}}(\pi, \gamma) = \text{sol}_{\{s\}}(\pi, \gamma)$.*

Proof. Assume w.l.o.g. that $r \leq s$. In order to show that $\text{sol}_{\{r\}}(\pi, \gamma) \subseteq \text{sol}_{\{s\}}(\pi, \gamma)$, let $\varphi \in \text{sol}_{\{r\}}(\pi, \gamma)$. So we have $\pi^r \models \gamma[\varphi]$ and $\pi^s \models \gamma[\top]$, where $r \leq s$. Thus, if for all $r \leq i < s$ it holds that $\pi^i \not\models \gamma[\bot]$, we obtain by Lemma 5.2 that $\pi^s \models \gamma[\varphi]$, that is, $\varphi \in \text{sol}_{\{s\}}(\pi, \gamma)$. Hence, we know that $\text{sol}_{\{r\}}(\pi, \gamma) \subseteq \text{sol}_{\{s\}}(\pi, \gamma)$. On the other hand, in order to show that $\text{sol}_{\{s\}}(\pi, \gamma) \subseteq \text{sol}_{\{r\}}(\pi, \gamma)$, let $\varphi \in \text{sol}_{\{s\}}(\pi, \gamma)$. Further, let l be the cycle length of π according to Figure 5.2. Since $r \in \text{cycle}(\pi)$, we

know that $\pi(r) = \pi(r + m \cdot l)$ for all $m \in \mathbb{N}$. So we have $\pi^s \models \gamma[\varphi]$ and $\pi^{r+m \cdot l} \models \gamma[\top]$, where $s \leq r + m \cdot l$ for an appropriate large number m. Thus, if for all $s \leq i < r + m \cdot l$ it holds that $\pi^i \not\models \gamma[\bot]$, we obtain by Lemma 5.2 that $\pi^{r+m \cdot l} \models \gamma[\varphi]$, that is, $\varphi \in \mathrm{sol}_{\{r\}}(\pi, \gamma)$. Hence, we know that $\mathrm{sol}_{\{s\}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{r\}}(\pi, \gamma)$. □

Recall that if the highest index as required in Lemma 5.3 does not exist, we know that there are indices in $\mathcal{I}$ that refer to states in a cycle. But then, in order to eliminate existential choices, it suffices to choose any index that refers to a state in a cycle, since the solutions at these states are the same according to Lemma 5.4. This is shown in the following lemma.

Lemma 5.5. *Let γ be an intermediate collecting query, π be a path, and $\mathcal{I} \subseteq \mathbb{N}$. Suppose that there is a least index $m \in \mathcal{I}$ and an index $n \in \mathrm{cycle}(\pi) \cap \mathcal{I}$ such that $\pi^m \models \gamma[\top]$ and $\pi^n \models \gamma[\top]$. If $\pi^i \not\models \gamma[\bot]$ for all $i \geq m$, then $\mathrm{sol}_{\{n\}}(\pi, \gamma) = \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$.*

Proof. Since $n \in \mathcal{I}$, we know that $\mathrm{sol}_{\{n\}}(\pi, \gamma) \subseteq \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$. In order to show that $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{n\}}(\pi, \gamma)$, let $\varphi \in \mathrm{sol}_{\mathcal{I}}(\pi, \gamma)$. Therefore, we know that there exists $j \in \mathcal{I}$ such that $\pi^j \models \gamma[\varphi]$. Since m is the least index such that $\pi^m \models \gamma[\top]$, we know that $m \leq j$. If $j \leq n$, we have $\pi^j \models \gamma[\varphi]$ and $\pi^n \models \gamma[\top]$. Thus, if for all $j \leq i < n$ it holds that $\pi^i \not\models \gamma[\bot]$, we obtain by Lemma 5.2 that $\pi^n \models \gamma[\varphi]$, that is, $\varphi \in \mathrm{sol}_{\{n\}}(\pi, \gamma)$. Hence, we know that $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{n\}}(\pi, \gamma)$. Otherwise, if $j > n$, we know that $j \in \mathrm{cycle}(\pi)$ since $n \in \mathrm{cycle}(\pi)$ (cf. Figure 5.2). Moreover, by Lemma 2.1, we know that $\pi^j \models \gamma[\top]$. So we have $n, j \in \mathrm{cycle}(\pi)$ and $\pi^n \models \gamma[\top]$ as well as $\pi^j \models \gamma[\top]$. Thus, if for all $i \in \mathrm{cycle}(\pi)$ it holds that $\pi^i \not\models \gamma[\bot]$, we obtain by Lemma 5.4 that $\mathrm{sol}_{\{n\}}(\pi, \gamma) = \mathrm{sol}_{\{j\}}(\pi, \gamma)$, that is, $\varphi \in \mathrm{sol}_{\{n\}}(\pi, \gamma)$. Hence, we know again that $\mathrm{sol}_{\mathcal{I}}(\pi, \gamma) \subseteq \mathrm{sol}_{\{n\}}(\pi, \gamma)$. □

Note that instead of computing the solutions at a single state in the cycle according to Lemma 5.5, we could also compute the solutions at the set of all states with indices $i \in \mathrm{cycle}(\pi) \cap \mathcal{I}$ satisfying $\pi^i \models \gamma[\top]$. This is allowed since the solutions at these states are the same according to Lemma 5.4.

Let us now demonstrate how the above results can be applied on order to eliminate non-determinism when solving queries.

Example 5.4. Consider any path π and the query $\gamma = \theta \,\mathbf{U}\, \bar{\gamma}$, where $\bar{\gamma}$ is intermediate collecting. By Lemma 5.1, we know that for every formula φ

it holds that $\pi \models \gamma[\varphi]$ iff there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\varphi]$. Hence, in the worst case we have to check all indices in $\mathcal{I}_\pi(\theta)$ in order to determine whether some formula φ is a solution to γ on π.

Obviously, if there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\bot]$, then every formula is a solution to γ on π. Let us for the moment being assume that for all $i \in \mathcal{I}_\pi(\theta)$ it holds that $\pi^i \not\models \bar{\gamma}[\bot]$. Similarly, if for all $i \in \mathcal{I}_\pi(\theta)$ it holds that $\pi^i \not\models \bar{\gamma}[\top]$, then γ has no solution on π. Since this is also a special case, we assume that there exists at least one $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\top]$. Note that these assumptions enable us to apply Lemma 5.3 and Lemma 5.5. Although checking them seems to be complex, it can be easily implemented by symbolic algorithms as we will do in the following sections.

Now, we have to distinguish between two cases. If there exists a highest index $n \in \mathcal{I}_\pi(\theta)$ such that $\pi^n \models \bar{\gamma}[\top]$, we know by Lemma 5.3 that $\pi \models \gamma[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$. Otherwise, if no such highest index exists, then $\mathcal{I}_\pi(\theta) = \mathbb{N}$ and there exists $n \in \text{cycle}(\pi)$ such that $\pi^n \models \bar{\gamma}[\top]$. Thus, we know by Lemma 5.5 that $\pi \models \gamma[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$. Hence, in both cases, solving γ can be reduced to solving $\bar{\gamma}$ without existential choices.

In the following section, we will show how our results can be exploited in CTLQ^x query solving algorithms. At first, we start with a thorough analysis of Chan's algorithm. This algorithm is applicable to the valid subset of CTLQ^x, i.e., queries in CTLQ^x that have a solution in every Kripke structure. Afterwards, we present an extension of Chan's algorithm that is able to solve all queries in our fragment CTLQ^x.

5.2.2 The Chan Algorithm

The Chan algorithm was introduced by William Chan [Cha00] in order to solve queries in his syntactic fragment of *valid* CTL queries, i.e., CTL queries that are exact and always have a solution. However, Chan neither proved the correctness of his algorithm nor did he describe its functionality. In the author's diploma thesis [Sam02, SV03], it was shown that Chan's fragment of valid CTL queries is erroneous and a correct version was presented. It can be easily verified that the class of valid queries in CTLQ^x subsumes the fragment presented there. Note that valid queries within our fragment CTLQ^x can be simply obtained by restricting the grammar in Table 4.3 and Table 4.4 to those operators that guarantee validity according to

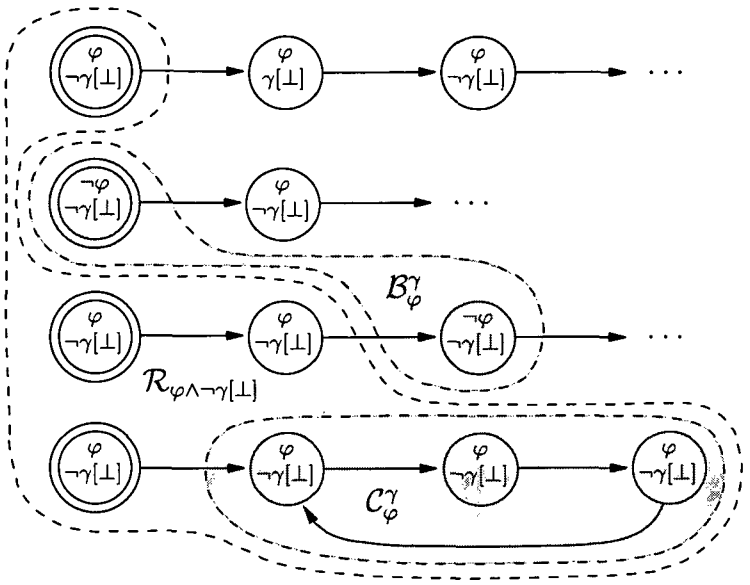


Figure 5.3: Auxiliary sets in Chan's algorithm

our observations in Section 2.5.1. We will now prove that all valid queries in CTLQ^x can be solved by Chan's symbolic algorithm (cf. Section 2.4.1).

Definition 5.5 (Auxiliary sets). Following Chan, we introduce the following three macros (parameterized by φ and γ) as abbreviations:

$$\mathcal{R}_{\varphi} = \mu\mathcal{Z}.((\mathcal{Q} \cup \mathrm{post}_{\exists}(\mathcal{Z})) \cap [\![\varphi]\!]) \qquad \textit{(Reachable set)}$$

$$\mathcal{C}^{\gamma}_{\varphi} = \nu\mathcal{Z}.(\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]} \cap \mathrm{post}_{\exists}(\mathcal{Z})) \qquad \textit{(Cycle set)}$$

$$\mathcal{B}^{\gamma}_{\varphi} = (\mathcal{Q} \cup \mathrm{post}_{\exists}(\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]})) \setminus ([\![\varphi]\!] \cup [\![\gamma[\bot]]\!]) \qquad \textit{(Boundary set)}$$

The intuitive meaning of these three auxiliary sets is illustrated in Figure 5.3, where the initial set $\mathcal{Q}$ is assumed to consist of the four double-circled states. The set $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$ consists of those states that are reachable from states in $\mathcal{Q}$ by going only through states at which $\varphi \wedge \neg\gamma[\bot]$ holds. In particular, $\mathcal{R}_{\top}$ consists of all states that are reachable from states in $\mathcal{Q}$. The set $\mathcal{C}^{\gamma}_{\varphi}$ consists of all states within a cycle in $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$. Finally, the set $\mathcal{B}^{\gamma}_{\varphi}$ consists of the boundary of $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$, i.e., the first states on each path starting from $\mathcal{Q}$ that are not in $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$ and at which $\gamma[\bot]$ does not hold. The following lemma states the meaning of these sets more formally.

Lemma 5.6. *Let γ be a query, φ be a formula, and $\mathcal{Q}$ be a set of states in a Kripke structure. Moreover, let $\Pi \subseteq \text{paths}(\mathcal{Q})$ such that $\pi \in \Pi$ iff $\pi^i \not\models \gamma[\bot]$ for all $i \in \mathcal{I}_\pi(\varphi)$. Then, $\mathcal{R}_\varphi$, $\mathcal{C}_\varphi^\gamma$, and $\mathcal{B}_\varphi^\gamma$ are the sets of states on paths $\pi \in \text{paths}(\mathcal{Q})$ such that*

1. *$\pi(n) \in \mathcal{R}_\varphi$ iff for all $i \leq n$ it holds that $\pi^i \models \varphi$.*
2. *$\pi(n) \in \mathcal{C}_\varphi^\gamma$ iff $\pi \in \Pi$, $\mathcal{I}_\pi(\varphi)$ is infinite, and $n \in \text{cycle}(\pi)$.*
3. *$\pi(n) \in \mathcal{B}_\varphi^\gamma$ iff $\pi \in \Pi$, $\mathcal{I}_\pi(\varphi)$ is finite, and $n = \max(\mathcal{I}_\pi(\varphi))$.*

Proof. We consider the three cases separately:

1. By the fixpoint definition of $\mathcal{R}_\varphi$, it is easy to see that starting from the states in $\mathcal{Q}$ that satisfy φ, only successor states are added that satisfy φ as well. This is repeated until a fixpoint is reached. Hence, it follows trivially that $\mathcal{R}_\varphi$ consists of exactly those states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ where $\pi^i \models \varphi$ for all $i \leq n$.

2. By the fixpoint definition of $\mathcal{C}_\varphi^\gamma$, it is easy to see that $\mathcal{C}_\varphi \subseteq \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$. In particular, $\mathcal{C}_\varphi^\gamma$ consists of exactly those states that are in a cycle that is entirely contained in $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$. By the definition of $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$, we know that a cycle on a path $\pi \in \text{paths}(\mathcal{Q})$ is entirely contained in $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$ iff $\pi^i \models \varphi \wedge \neg\gamma[\bot]$ for all $i \in \mathbb{N}$. Hence, since $\mathcal{I}_\pi(\varphi) = \mathbb{N}$ iff $\pi^i \models \varphi$ for all $i \in \mathbb{N}$, and $\pi \in \Pi$ iff $\pi^i \not\models \gamma[\bot]$ for all $i \in \mathcal{I}_\pi(\varphi)$, it follows that $\mathcal{C}_\varphi^\gamma$ consists of exactly those states $\pi(n)$ where $\pi \in \Pi$, $\mathcal{I}_\pi(\varphi)$ is infinite, and $n \in \text{cycle}(\pi)$.

3. By the definition of $\mathcal{B}_\varphi^\gamma$, it is easy to see that starting from $\mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$, the set of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ is computed where for all $i < n$ it holds that $\pi^i \models \varphi \wedge \neg\gamma[\bot]$. Then, all states satisfying φ or $\gamma[\bot]$ are removed. Therefore, the remaining set consists of those states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ where for all $i < n$ it holds that $\pi^i \models \varphi \wedge \neg\gamma[\bot]$ and $\pi^n \models \neg\varphi \wedge \neg\gamma[\bot]$. Since $\mathcal{I}_\pi(\varphi)$ is finite with $n = \max(\mathcal{I}_\pi(\varphi))$ iff $\pi^n \not\models \varphi$ and $\pi^i \models \varphi$ for all $i < n$, we know that $\mathcal{I}_\pi(\varphi) = \{i \in \mathbb{N} \mid i \leq n\}$. Hence, since $\pi \in \Pi$ iff $\pi^i \not\models \gamma[\bot]$ for all $i \in \mathcal{I}_\pi(\varphi)$, it follows that $\mathcal{B}_\varphi^\gamma$ consists of exactly those states $\pi(n)$ where $\pi \in \Pi$, $\mathcal{I}_\pi(\varphi)$ is finite, and $n = \max(\mathcal{I}_\pi(\varphi))$.

This concludes the proof. □

Algorithm 1 Chan algorithm

```
1  function ExactSol(γ, Q) begin
2    case γ of
3      ? :          return Q ;
4      θ ∨ γ̄ :      return ExactSol(γ̄, Q \ [[θ]]) ;
5      AX γ̄ :       return ExactSol(γ̄, post∃(Q)) ;
6      AF γ̄ :       return ExactSol(A(⊤ U γ̄), Q) ;
7      AG γ̄ :       return ExactSol(A(γ̄ Ŵ ⊥), Q) ;
8      A(θ U γ̄) :   return ExactSol(γ̄, B_θ^γ̄ ∪ C_θ^γ̄) ;
9      A(γ̄ W θ) :   return ExactSol(A((θ ∨ γ̄) Ŵ θ), Q) ;
10     A(γ̄ Ŵ θ) :   return ExactSol(γ̄, Q ∪ post∃(R_¬θ)) ;
11     A(θ W γ̄) :   return ExactSol(γ̄, B_θ^γ̄) ;
12     A(θ W̄ γ̄) :   return ExactSol(γ̄, (Q ∪ post∃(R_θ)) \ [[θ]]) ;
13   esac
14 end
```

Remark 5.2. Note that the states in $\mathcal{B}_\varphi$ are those with the highest index on each path satisfying the conditions of Lemma 5.3, and the states in $\mathcal{C}_\varphi$ are those within a cycle on each path satisfying the conditions of Lemma 5.5. Moreover, note that the assumptions on states satisfying $\gamma[\top]$ in these lemmas can currently be ignored since the queries we consider are valid, i.e., these assumptions are always satisfied. Therefore, by using the above auxiliary sets, we are able to define a query solving algorithm that implements the kind of determinization described in the previous section.

Chan's algorithm for solving valid queries in CTLQ^x is shown in Algorithm 1. In order to prove its correctness, we need the following definition. Recall from Section 2.3.2 that a CTL formula holds at a set of states in a Kripke structure iff it holds at each state in this set.

Definition 5.6 (Solution states). Let γ be a query and $\mathcal{Q}$ be a set of states in a Kripke structure. A set of states $\mathcal{S}$ is a set of *solution states* to γ at $\mathcal{Q}$ iff it holds that $\mathcal{S} \models \varphi$ implies $\mathcal{Q} \models \gamma[\varphi]$ for all formulas φ. A set of solution states $\mathcal{S}$ is *unique* (or simply *the* set of solution states) to γ at $\mathcal{Q}$ iff it holds that $\mathcal{Q} \models \gamma[\varphi]$ implies $\mathcal{S} \models \varphi$ for all formulas φ.

It is easy to see that for every solution φ to a query γ at a set of states $\mathcal{Q}$

there exists a set of solution states $\mathcal{S}$ to γ at $\mathcal{Q}$ such that $\mathcal{S} \models \varphi$. Moreover, note that a unique set of solution states is indeed *unique* if all states in the Kripke structure are labeled differently. Otherwise, there may exist several unique sets of solution states. However, since all these sets of solution states must represent the same solutions by definition, they are equivalent. The following proposition shows that a query is exact if and only if there always exists a unique set of solution states to this query.

Proposition 5.2. *A query γ is exact iff for every set of states $\mathcal{Q}$ in a Kripke structure satisfying $\mathcal{Q} \models \gamma[\top]$ there exists a unique set of solution states.*

Proof. For the *if* direction, assume that there exists a unique set of solution states $\mathcal{S}$ to γ at any set of states $\mathcal{Q}$, i.e., $\mathcal{Q} \models \gamma[\varphi]$ iff $\mathcal{S} \models \varphi$ for all formulas φ. Now, let ξ be the conjunction of all formulas φ satisfying $\mathcal{S} \models \varphi$. Then, it obviously holds that $\mathcal{S} \models \xi$ and $\mathcal{S} \models \varphi$ iff $\xi \Rightarrow \varphi$. Thus, by definition, we know that $\mathcal{Q} \models \gamma[\xi]$ and $\mathcal{Q} \models \gamma[\varphi]$ iff $\xi \Rightarrow \varphi$, i.e., ξ is an exact solution to γ at $\mathcal{Q}$. Hence, since $\mathcal{Q}$ was chosen w.l.o.g., γ is exact.

For the *only if* direction, assume that γ is exact and let $\mathcal{Q}$ be any set of states. If γ has no solution at $\mathcal{Q}$, we know by Lemma 2.1 that $\mathcal{Q} \not\models \gamma[\top]$ and we are done. Otherwise, since γ is exact, there must exist an exact solution ξ and therefore a set of solution states $\mathcal{S}$ to γ at $\mathcal{Q}$ such that $\mathcal{S} \models \xi$. Thus, since $\mathcal{Q} \models \gamma[\varphi]$ iff $\xi \Rightarrow \varphi$, we know that $\mathcal{Q} \models \gamma[\varphi]$ implies $\mathcal{S} \models \varphi$ for all formulas φ. Hence, $\mathcal{S}$ is a unique set of solution states to γ at $\mathcal{Q}$.

We will now show that if all states in the Kripke structure are labelled differently, then there exists no other unique set of solution states. To this aim, let $\mathcal{S}'$ be any unique set of solution states to γ at $\mathcal{Q}$. Now, let $\psi_{\mathcal{S}}$ and $\psi_{\mathcal{S}'}$ be the characteristic functions of $\mathcal{S}$ and $\mathcal{S}'$ respectively. Then, we trivially have $\mathcal{S} \models \psi_{\mathcal{S}}$ and $\mathcal{S}' \models \psi_{\mathcal{S}'}$. Hence, since $\mathcal{S}$ and $\mathcal{S}'$ are sets of solution states, we obtain $\mathcal{Q} \models \gamma[\psi_{\mathcal{S}}]$ and $\mathcal{Q} \models \gamma[\psi_{\mathcal{S}'}]$. Thus, since $\mathcal{Q} \models \gamma[\psi_{\mathcal{S}}]$ implies $\mathcal{S}' \models \psi_{\mathcal{S}}$, we know that $\mathcal{S}' \subseteq \mathcal{S}$. Analogously, since $\mathcal{Q} \models \gamma[\psi_{\mathcal{S}'}]$ implies $\mathcal{S} \models \psi_{\mathcal{S}'}$, we know that $\mathcal{S} \subseteq \mathcal{S}'$. Hence, $\mathcal{S} = \mathcal{S}'$. □

Now, we are able to prove the correctness of Chan's algorithm. Note that for simplicity we will write *set of solution states* to denote the *unique set of solution states*. This should not be confusing since we are only interested in the unique set of solution states to exact queries.

Theorem 5.1. *Let $\gamma \in CTLQ^x$ be valid and $\mathcal{Q}$ be a set of states in a Kripke structure. Moreover, let* ExactSol *be the function defined in Algorithm 1. Then,* $\text{ExactSol}(\gamma, \mathcal{Q})$ *returns the unique set of solution states to γ at $\mathcal{Q}$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = ?$. Then, the assertion holds trivially since $\text{ExactSol}(?, \mathcal{Q}) = \mathcal{Q}$ and $\mathcal{Q}$ is the unique set of solution states to ? at $\mathcal{Q}$.

Induction step:

▷ Let $\gamma = \theta \vee \bar{\gamma}$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{Q} \setminus [\![\theta]\!])$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{Q} \setminus [\![\theta]\!]$. Therefore, we have $\mathcal{Q} \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Moreover, note that trivially $\mathcal{Q} \cap [\![\theta]\!] \models \theta$. Hence, since $\mathcal{Q} \models \theta \vee \bar{\gamma}[\varphi]$ iff $\mathcal{Q} \cap [\![\theta]\!] \models \theta$ and $\mathcal{Q} \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$, we know that $\mathcal{Q} \models \theta \vee \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \text{post}_\exists(\mathcal{Q}))$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\text{post}_\exists(\mathcal{Q})$. Therefore, we have $\text{post}_\exists(\mathcal{Q}) \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Hence, since $\mathcal{Q} \models \mathbf{AX}\,\bar{\gamma}[\varphi]$ iff $\text{post}_\exists(\mathcal{Q}) \models \bar{\gamma}[\varphi]$, we know that $\mathcal{Q} \models \mathbf{AX}\,\bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$. Since $\mathbf{AF}\,\bar{\gamma} \Leftrightarrow \mathbf{A}(\top\,\mathbf{U}\,\bar{\gamma})$, the assertion follows from the case of the strong until operator $\mathbf{U}$.

▷ Let $\gamma = \mathbf{AG}\,\bar{\gamma}$. Since $\mathbf{AG}\,\bar{\gamma} \Leftrightarrow \mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\bot)$ and $\mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\bot) \Leftrightarrow \mathbf{A}((\bot \vee \bar{\gamma})\,\mathring{\mathbf{W}}\,\bot)$, we know that $\mathbf{AG}\,\bar{\gamma} \Leftrightarrow \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{W}}\,\bot)$. Hence, the assertion follows from the case of the overlapping weak until operator $\mathring{\mathbf{W}}$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma})$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{B}_\theta^{\bar{\gamma}} \cup \mathcal{C}_\theta^{\bar{\gamma}})$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{B}_\theta^{\bar{\gamma}} \cup \mathcal{C}_\theta^{\bar{\gamma}}$. Therefore, we have $\mathcal{B}_\theta^{\bar{\gamma}} \cup \mathcal{C}_\theta^{\bar{\gamma}} \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma}[\varphi])$, i.e., for each $\pi \in \text{paths}(\mathcal{Q})$ it holds that $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$.
By Lemma 5.1, we know that $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$ iff there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\varphi]$, that is, $\varphi \in \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. In particular, if there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\bot]$, then $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$ is trivially true. Hence, it suffices to consider only paths where $\pi^i \not\models \bar{\gamma}[\bot]$ for all $i \in \mathcal{I}_\pi(\theta)$. Thus,

let $\Pi \subseteq \text{paths}(\mathcal{Q})$ be the set of such paths, and for each $\pi \in \Pi$ let us distinguish between two cases:
(i) $\mathcal{I}_\pi(\theta)$ is infinite: Since γ is valid, we know that also $\bar{\gamma}$ is valid, which implies $\pi^i \models \bar{\gamma}[\top]$ for all $i \in \text{cycle}(\pi)$. Thus, by Proposition 5.1 and Lemma 5.5, we know that $\text{sol}_{\{i\}}(\pi, \bar{\gamma}) = \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$ for every $i \in \text{cycle}(\pi)$. Hence, since all states with indices in $\text{cycle}(\pi)$ have the same solutions, this implies that $\pi \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi]$ iff $\pi^{\text{cycle}(\pi)} \models \bar{\gamma}[\varphi]$.
(ii) $\mathcal{I}_\pi(\theta)$ is finite: Since γ is valid, we know that also $\bar{\gamma}$ is valid, which implies $\pi^i \models \bar{\gamma}[\top]$ for all $i \in \mathcal{I}_\pi(\theta)$. In particular, since $\mathcal{I}_\pi(\theta)$ is finite, this implies that there exists $n = \max(\mathcal{I}_\pi(\theta))$ satisfying $\pi^n \models \bar{\gamma}[\top]$. Thus, by Proposition 5.1 and Lemma 5.3, we obtain $\text{sol}_{\{n\}}(\pi, \bar{\gamma}) = \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. Consequently, $\pi \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$.
By Lemma 5.6, we know that $\mathcal{C}_\theta^{\bar{\gamma}}$ consists of all states within cycles according to item (i) on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\theta)$ is infinite, and $\mathcal{B}_\theta^{\bar{\gamma}}$ consists of all states with index $n = \max(\mathcal{I}_\pi(\theta))$ according to item (ii) on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\theta)$ is finite. Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\theta \,\mathbf{U}\, \bar{\gamma}[\varphi])$ iff $\mathcal{B}_\theta^{\bar{\gamma}} \cup \mathcal{C}_\theta^{\bar{\gamma}} \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\theta \,\mathbf{U}\, \bar{\gamma}[\varphi])$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathbf{W}\, \theta)$. Since $\mathbf{A}(\bar{\gamma} \,\mathbf{W}\, \theta) \Leftrightarrow \mathbf{A}((\theta \vee \bar{\gamma}) \,\mathring{\mathbf{W}}\, \theta)$, the assertion follows from the cases of the overlapping weak until operator $\mathring{\mathbf{W}}$ and disjunction.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathring{\mathbf{W}}\, \theta) = \mathbf{A}((\mathbf{G}\, \bar{\gamma}) \vee (\bar{\gamma} \,\mathring{\mathbf{U}}\, \theta))$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta}))$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta})$. Therefore, we have $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta}) \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta)$, i.e., for each $\pi \in \text{paths}(\mathcal{Q})$ it holds that $\pi \models \bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta$. Let us distinguish between two cases:
(i) If $\pi^i \not\models \theta$ for all $i \in \mathbb{N}$, we know that $\pi \models \bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta$ iff $\pi \models \mathbf{G}\, \bar{\gamma}[\varphi]$.
(ii) Otherwise, if there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \theta$, it is easy to see that $\pi \models \bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta$ iff $\pi^i \models \bar{\gamma}[\varphi]$ for all $i \leq n$.
By Lemma 5.6, we know that $\mathcal{R}_{\neg\theta}$ consists of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ such that $\pi^i \not\models \theta$ for all $i \leq n$. Hence, $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta})$ consists of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ such that $\pi^i \not\models \theta$ for all $i < n$. Thus, it is easy to see that $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta})$ consists of all states on paths $\pi \in \text{paths}(\mathcal{Q})$ satisfying the condition of item (i), and all states $\pi(i)$ with $i \leq n$, where n is the least number such that

$\pi^n \models \theta$, on paths $\pi \in \text{paths}(\mathcal{Q})$ satisfying the condition of item (ii). Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\bar{\gamma}[\varphi]\,\mathring{\mathbf{W}}\,\theta)$ iff $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta}) \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\bar{\gamma}[\varphi]\,\mathring{\mathbf{W}}\,\theta)$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,\bar{\gamma}))$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{B}_\theta^{\bar{\gamma}})$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{B}_\theta^{\bar{\gamma}}$. Therefore, we have $\mathcal{B}_\theta^{\bar{\gamma}} \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}[\varphi])$, i.e., for each $\pi \in \text{paths}(\mathcal{Q})$ it holds that $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$.
By Lemma 5.1, we know that if $\pi \not\models \mathbf{G}\,\theta$, it holds that $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$ iff there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\varphi]$, that is, $\varphi \in \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. In particular, if there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\bot]$, then $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$ is trivially true. Also in the case of $\pi \models \mathbf{G}\,\theta$, we trivially obtain $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$. Hence, since $\pi \not\models \mathbf{G}\,\theta$ iff $\mathcal{I}_\pi(\theta)$ is finite, it suffices to consider only paths where $\mathcal{I}_\pi(\theta)$ is finite and $\pi^i \not\models \bar{\gamma}[\bot]$ for all $i \in \mathcal{I}_\pi(\theta)$. Thus, let $\Pi \subseteq \text{paths}(\mathcal{Q})$ be the set of such paths, and $\pi \in \Pi$.
Since γ is valid, we know that also $\bar{\gamma}$ is valid, which implies $\pi^i \models \bar{\gamma}[\top]$ for all $i \in \mathcal{I}_\pi(\theta)$. In particular, since $\mathcal{I}_\pi(\theta)$ is finite, this implies that there exists $n = \max(\mathcal{I}_\pi(\theta))$ satisfying $\pi^n \models \bar{\gamma}[\top]$. Thus, by Proposition 5.1 and Lemma 5.3, we obtain $\text{sol}_{\{n\}}(\pi, \bar{\gamma}) = \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. Consequently, $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$.
By Lemma 5.6, we know that $\mathcal{B}_\theta^{\bar{\gamma}}$ consists of all states $\pi(n)$ on paths $\pi \in \Pi$, where $n = \max(\mathcal{I}_\pi(\theta))$. Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}[\varphi])$ iff $\mathcal{B}_\theta^{\bar{\gamma}} \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}[\varphi])$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\bar{\mathbf{U}}\,\bar{\gamma}))$ By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, (\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!])$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $(\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!]$. Therefore, we have $(\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi])$, i.e., for each $\pi \in \text{paths}(\mathcal{Q})$ it holds that $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi]$.
If $\pi \models \mathbf{G}\,\theta$, then $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi]$ is trivially true. Hence, it suffices to consider only paths where $\pi \not\models \mathbf{G}\,\theta$. Thus, let $\Pi \subseteq \text{paths}(\mathcal{Q})$ be the set of such paths, and $\pi \in \Pi$. Then, there exists a least $n \in \mathbb{N}$ such that $\pi^n \not\models \theta$. It is easy to see that $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$.
By Lemma 5.6, we know that $\mathcal{R}_\theta$ consists of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ such that $\pi^i \models \theta$ for all $i \leq n$. Hence, $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_\theta)$

	$\mathcal{R}_\varphi$	$\mathcal{C}^\gamma_\varphi$	$\mathcal{B}^\gamma_\varphi$
$\varphi = \top$	$\mathcal{R}_\top$	$\mathcal{C}^\gamma_\top$	$\emptyset$
$\varphi = \bot$	$\emptyset$	$\emptyset$	$\mathcal{Q} \setminus [\![\gamma[\bot]]\!]$

Table 5.2: Auxiliary sets with constant truth values

consists of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ such that $\pi^i \models \theta$ for all $i < n$. Thus, it is easy to see that $(\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!]$ consists of all states $\pi(n)$ on paths $\pi \in \Pi$, where n is the least number such that $\pi^n \not\models \theta$. Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\theta\, \bar{\mathbf{W}}\, \bar{\gamma}[\varphi])$ iff $(\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\theta\, \bar{\mathbf{W}}\, \bar{\gamma}[\varphi])$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

This concludes the proof. □

Remark 5.3. Note that the Chan algorithm can be improved in several ways. For example, if ExactSol$(\gamma, \emptyset)$ is called for any query γ, the sequence of recursive calls can be aborted since the result will be $\emptyset$ independently of any further recursive calls. Moreover, the recursive calls for the cases $\theta \vee \bar{\gamma}$, $\mathbf{A}(\theta\, \mathbf{U}\, \bar{\gamma})$, $\mathbf{A}(\bar{\gamma}\, \mathring{\mathbf{W}}\, \theta)$, $\mathbf{A}(\theta\, \mathbf{W}\, \bar{\gamma})$, and $\mathbf{A}(\theta\, \bar{\mathbf{W}}\, \bar{\gamma})$ can be simplified if $\theta \in \{\top, \bot\}$. This follows immediately from the definition of the auxiliary sets in Chan's algorithm as shown in Table 5.2.

Obviously, in order to obtain an exact solution, we have to compute the strongest formula (within the set of interesting solutions) that holds at the unique set of solution states. In particular, if we are interested in propositional solutions, the following definition is of interest.

Definition 5.7 (Characteristic function). Let $\mathfrak{K}$ be a Kripke structure over the set of atomic propositions $\mathcal{A}$. The *characteristic function* of a set of states $\mathcal{Q}$ in $\mathfrak{K}$ is given by $\bigvee_{s \in \mathcal{Q}} (\bigwedge_{p \in \ell(s)} p \wedge \bigwedge_{p \in \mathcal{A} \setminus \ell(s)} \neg p)$.

Now, the following corollary shows how to obtain a propositional exact solution by Chan's algorithm. Of course, propositional solutions can be further restricted to propositional solutions consisting of interesting atomic propositions in order to decrease the computational effort.

Corollary 5.1. *Let $\gamma \in CTLQ^x$ be valid and s_0 be the initial state of a Kripke structure $\mathfrak{K}$. Moreover, let* ExactSol *be the function defined in Algorithm 1. Then, the characteristic function of the set returned by* ExactSol(γ, $\{s_0\}$) *is a propositional exact solution to γ in $\mathfrak{K}$.*

Proof. Let $\mathcal{Q}$ be the set returned by ExactSol(γ, $\{s_0\}$) and ξ be the characteristic function of $\mathcal{Q}$. It is then easy to see that $\mathcal{Q} \models \xi$. By Theorem 5.1, this implies $s_0 \models \gamma[\xi]$, i.e., ξ is a solution to γ in $\mathfrak{K}$. Now, let φ be any propositional solution to γ in $\mathfrak{K}$, that is, $s_0 \models \gamma[\varphi]$. Hence, by Theorem 5.1, we know that $\mathcal{Q} \models \varphi$. W.l.o.g., it can be assumed that φ is in conjunctive normal form, that is, $\varphi = \varphi_1 \wedge \ldots \wedge \varphi_n$. Then, w.l.o.g., we choose any conjunct $\varphi_i = \alpha_{i,1} \vee \ldots \vee \alpha_{i,m}$ of φ. Since $\mathcal{Q} \models \varphi$, it follows that $\mathcal{Q} \models \varphi_i$ and therefore $\mathcal{Q} \models \alpha_{i,j}$ for some $1 \leq j \leq m$. This means that for each $s \in \mathcal{Q}$ there exists $1 \leq j \leq m$ such that $\alpha_{i,j} \in \ell(s)$ or $\neg\alpha_{i,j} \in \mathcal{A} \setminus \ell(s)$. Thus, since φ_i was chosen w.l.o.g., we know that $\xi \Rightarrow \varphi$. Hence, ξ is a propositional exact solution to γ in $\mathfrak{K}$. □

Since the Chan algorithm is restricted to the valid subset of CTLQ^x, we will now show how it can be extended in order to solve all queries in CTLQ^x.

5.2.3 The Extended Chan Algorithm

As shown in the previous section, Chan's algorithm is based on the assumption that solving a query can be reduced to solving its subqueries without existential choices. Although this kind of determinization is not possible in general, it can be performed for the whole fragment CTLQ^x. Recall that by Lemma 5.3 and Lemma 5.5, solving an intermediate collecting query γ at several states, whose indices on a path are given by a set $\mathcal{I}$, can be reduced to solving γ at a single state with index in $\mathcal{I}$. In particular, if there exists a highest index in $\mathcal{I}$ such that γ has a solution at the corresponding state, then it suffices to solve γ at this single state. Otherwise, if no such highest index in $\mathcal{I}$ exists, then it suffices to solve γ at any state with index in $\mathcal{I}$ that is in a cycle and at which γ has a solution. Since the Chan algorithm is restricted to valid queries, such highest indices resp. cycle indices can be easily found because valid queries are guaranteed to have a solution at every state. Hence, we can simply choose the maximum of $\mathcal{I}$, if it exists, or, otherwise, any cycle index in $\mathcal{I}$.

However, when extending Chan's algorithm (see Algorithm 2) to the whole fragment CTLQ^x, validity is no longer guaranteed. Therefore, the required states have to be computed in a more sophisticated way. In particular, we compute the set of states that correspond to highest indices resp. cycle indices in $\mathcal{I}$. Afterwards, we traverse the corresponding paths backwards until a state on each path is found at which γ has a solution. These are then the states with highest indices among the states at which γ has a solution, i.e., it suffices to solve γ at this uniquely determined set of states. This idea of computing the states at which γ has a solution and that are furthest away is implemented by the function FurthestSol in Algorithm 2.

The following lemma states the correctness of this function when the fourth argument is set to *false*, i.e., when paths are ignored whose corresponding set of prefix indices is infinite.

Lemma 5.7. *Let $\gamma \in$ CTLQ^x, φ be a formula, $\mathcal{Q}$ be a set of states in a Kripke structure, and* FurthestSol *be the function defined in Algorithm 2. Moreover, let $\Pi \subseteq \text{paths}(\mathcal{Q})$ such that $\pi \in \Pi$ iff $\mathcal{I}_\pi(\varphi)$ is finite, $\pi^i \models \gamma[\top]$ for some $i \in \mathcal{I}_\pi(\varphi)$, and $\pi^i \not\models \gamma[\bot]$ for all $i \in \mathcal{I}_\pi(\varphi)$. Then,* FurthestSol$(\gamma, \varphi, \mathcal{Q}, \mathit{false})$ *returns the set of states $\mathcal{S}$ on paths $\pi \in \Pi$ such that $\pi(n) \in \mathcal{S}$ iff n is the highest index in $\mathcal{I}_\pi(\varphi)$ satisfying $\pi^n \models \gamma[\top]$.*

Proof. Since *cycle* is false, the set $\mathcal{C}$ is empty. It is easy to see that the fixpoint $\mathcal{U}_1$ is then also empty. Consequently, the set $\mathcal{U}_2$ consists of the states in $\mathcal{B}^\gamma_\varphi \setminus [\![\gamma[\top]]\!]$, i.e., those states in the boundary set $\mathcal{B}^\gamma_\varphi$ at which γ is unsatisfiable. More formally, by Lemma 5.6 it follows that $\mathcal{U}_2$ is the set of states $\pi(n)$ on paths $\pi \in \Pi$, where $n = \max(\mathcal{I}_\pi(\varphi))$ and $\pi^n \not\models \gamma[\top]$.
Our aim is now to find the states with highest indices in $\mathcal{I}_\pi(\varphi)$ on the corresponding paths at which γ is satisfiable. Thus, starting at the set $\mathcal{U}_2$, we traverse the paths $\pi \in \Pi$ backwards in order to find the states with least indices $n \in \mathcal{I}_\pi(\varphi)$ such that $\pi^i \not\models \gamma[\top]$ for all $i \in \mathcal{I}_\pi(\varphi)$ with $i \geq n$. This is performed by the fixpoint computation of $\mathcal{U}_3$, i.e., the set $\mathcal{U}_3$ consists of the states $\pi(n)$ on paths $\pi \in \Pi$ where $\pi^i \not\models \gamma[\top]$ for all $i \in \mathcal{I}_\pi(\varphi)$ with $i \geq n$. Hence, it is easy to see that $\text{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$ is the set of states $\pi(n)$ on paths $\pi \in \Pi$, where $\pi^i \not\models \gamma[\top]$ for all $i \in \mathcal{I}_\pi(\varphi)$ with $i > n$ and there exists $i \in \mathcal{I}_\pi(\varphi)$ such that $i > n$.
Recall that, by Lemma 5.6, $\mathcal{B}^\gamma_\varphi$ consists of the states $\pi(n)$ on paths $\pi \in \Pi$, where $n = \max(\mathcal{I}_\pi(\varphi))$. Now, let $\mathcal{S} = ((\text{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}) \cup \mathcal{B}^\gamma_\varphi) \cap [\![\gamma[\top]]\!]$ and $\pi \in \Pi$. (i) If n is the highest index in $\mathcal{I}_\pi(\varphi)$ such that $\pi^n \models \gamma[\top]$

Algorithm 2 Extended Chan algorithm

```
1  function FurthestSol(γ, φ, 𝒬, cycle) begin
2    if cycle then 𝒞 = 𝒞_φ^γ else 𝒞 = ∅ ;
3    𝒰_1 = νZ.((𝒞 \ [[γ[⊤]]]) ∩ post_∃(Z)) ;
4    𝒰_2 = 𝒰_1 ∪ (ℬ_φ^γ \ [[γ[⊤]]]) ;
5    𝒰_3 = μZ.(((𝒰_2 ∪ pre_∃(Z)) ∩ ℛ_{φ∧¬γ[⊥]}) \ [[γ[⊤]]]) ;
6    return ((pre_∃(𝒰_3) ∩ ℛ_{φ∧¬γ[⊥]}) ∪ ℬ_φ^γ ∪ 𝒞) ∩ [[γ[⊤]]] ;
7  end
8
9  function EExactSol(γ, 𝒬) begin
10   case γ of
11     ? :          return 𝒬 ;
12     θ ∧ γ̄ :      return EExactSol(γ̄, 𝒬) ;
13     θ ∨ γ̄ :      return EExactSol(γ̄, 𝒬 \ [[θ]]) ;
14     AX γ̄ :       return EExactSol(γ̄, post_∃(𝒬)) ;
15     AF γ̄ :       return EExactSol(A(⊤ U γ̄), 𝒬) ;
16     AG γ̄ :       return EExactSol(A(γ̄ Ŵ ⊥), 𝒬) ;
17     A(γ̄ U θ) :   return EExactSol(A((θ ∨ γ̄) Ŵ θ), 𝒬) ;
18     A(γ̄ Ů θ) :   return EExactSol(A(γ̄ Ŵ θ), 𝒬) ;
19     A(θ U γ̄) :   return EExactSol(γ̄, FurthestSol(γ̄, θ, 𝒬, true)) ;
20     A(θ Ů γ̄) :   return EExactSol(A(θ U (θ ∧ γ̄)), 𝒬) ;
21     A(θ Ū γ̄) :   return EExactSol(A(θ W̄ γ̄), 𝒬) ;
22     A(γ̄ W θ) :   return EExactSol(A((θ ∨ γ̄) Ŵ θ), 𝒬) ;
23     A(γ̄ Ŵ θ) :   return EExactSol(γ̄, 𝒬 ∪ post_∃(ℛ_{¬θ})) ;
24     A(θ W γ̄) :   return EExactSol(γ̄, FurthestSol(γ̄, θ, 𝒬, false)) ;
25     A(θ Ŵ γ̄) :   return EExactSol(A(θ W (θ ∧ γ̄)), 𝒬) ;
26     A(θ W̄ γ̄) :   return EExactSol(γ̄, (𝒬 ∪ post_∃(ℛ_θ)) \ [[θ]]) ;
27   esac
28 end
```

and $n = \max(\mathcal{I}_\pi(\varphi))$, we know that $\pi(n) \in \mathcal{S}$ since $\pi(n) \in \mathcal{B}^\gamma_\varphi \cap [\![\gamma[\top]]\!]$. (ii) Otherwise, if n is the highest index in $\mathcal{I}_\pi(\varphi)$ such that $\pi^n \models \gamma[\top]$ and $n \neq \max(\mathcal{I}_\pi(\varphi))$, we know that $\pi(n) \in \mathcal{S}$ since $\pi(n) \in \mathrm{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]} \cap [\![\gamma[\top]]\!]$. It is easy to see that $\mathcal{S}$ contains no other states. Hence, the assertion follows by combining item (i) and item (ii). □

The following lemma states the correctness of function FurthestSol in Algorithm 2 when the fourth argument is set to *true*, i.e., when also paths are considered whose corresponding set of prefix indices is infinite.

Lemma 5.8. *Let $\gamma \in CTLQ^x$, φ be a formula, $\mathcal{Q}$ be a set of states in a Kripke structure, and* FurthestSol *be the function defined in Algorithm 2. Moreover, let $\Pi \subseteq \mathrm{paths}(\mathcal{Q})$ such that $\pi \in \Pi$ iff $\pi^i \models \gamma[\top]$ for some $i \in \mathcal{I}_\pi(\varphi)$ and $\pi^i \not\models \gamma[\bot]$ for all $i \in \mathcal{I}_\pi(\varphi)$. Then,* FurthestSol$(\gamma, \varphi, \mathcal{Q}, \mathit{true})$ *returns the set of states $\mathcal{S}$ on paths $\pi \in \Pi$ such that $\pi(n) \in \mathcal{S}$ iff*

- *If $\mathcal{I}_\pi(\varphi)$ is infinite and $\pi^i \models \gamma[\top]$ for some $i \in \mathrm{cycle}(\pi)$:*
  $n \in \mathrm{cycle}(\pi)$ and $\pi^n \models \gamma[\top]$
- *If $\mathcal{I}_\pi(\varphi)$ is finite or $\pi^i \not\models \gamma[\top]$ for all $i \in \mathrm{cycle}(\pi)$:*
  n is the highest index in $\mathcal{I}_\pi(\varphi)$ satisfying $\pi^n \models \gamma[\top]$

Proof. Since *cycle* is true, $\mathcal{C}$ denotes the set $\mathcal{C}^\gamma_\varphi$ of cycles on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\varphi)$ is infinite. In other words, by Lemma 5.6, this is the set of states $\pi(n)$ on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\varphi)$ is infinite and $n \in \mathrm{cycle}(\pi)$. Obviously, $\mathcal{C} \setminus [\![\gamma[\top]]\!]$ denotes the set of states in $\mathcal{C}$ at which γ is unsatisfiable, i.e., at which γ has no solution. It is easy to see that the greatest fixpoint $\mathcal{U}_1$ consists thus of those cycles in $\mathcal{C}$ that are entirely contained in $\mathcal{C} \setminus [\![\gamma[\top]]\!]$. More formally, $\mathcal{U}_1$ is the set of states $\pi(n)$ on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\varphi)$ is infinite, $\pi^i \not\models \gamma[\top]$ for all $i \in \mathrm{cycle}(\pi)$, and $n \in \mathrm{cycle}(\pi)$.
The set $\mathcal{U}_2$ is then defined as the union of the states in $\mathcal{U}_1$ and, in addition, of the states in $\mathcal{B}^\gamma_\varphi \setminus [\![\gamma[\top]]\!]$, i.e., those states in the boundary set $\mathcal{B}^\gamma_\varphi$ at which γ is unsatisfiable. More formally, by Lemma 5.6 it follows that $\mathcal{B}^\gamma_\varphi \setminus [\![\gamma[\top]]\!]$ is the set of states $\pi(n)$ on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\varphi)$ is finite, $n = \max(\mathcal{I}_\pi(\varphi))$, and $\pi^n \not\models \gamma[\top]$. Note that γ is unsatisfiable at all states in $\mathcal{U}_2$.
Our aim is now to find the states with highest indices in $\mathcal{I}_\pi(\varphi)$ on the corresponding paths at which γ is satisfiable. Thus, starting at the set $\mathcal{U}_2$, we traverse the paths $\pi \in \Pi$ backwards in order to find the states with least indices $n \in \mathcal{I}_\pi(\varphi)$ such that $\pi^i \not\models \gamma[\top]$ for all $i \in \mathcal{I}_\pi(\varphi)$ with $i \geq n$. This

is performed by the fixpoint computation of $\mathcal{U}_3$, i.e., the set $\mathcal{U}_3$ consists of the states $\pi(n)$ on paths $\pi \in \Pi$, where $\pi^i \not\models \gamma[\top]$ for all $i \in \mathcal{I}_\pi(\varphi)$ such that $i \geq n$. Hence, it is easy to see that $\text{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}$ is the set of states $\pi(n)$ on paths $\pi \in \Pi$, where $\pi^i \not\models \gamma[\top]$ for all $i \in \mathcal{I}_\pi(\varphi)$ with $i > n$ and there exists $i \in \mathcal{I}_\pi(\varphi)$ with $i > n$.
Recall that, by Lemma 5.6, $\mathcal{B}^\gamma_\varphi$ consists of the states $\pi(n)$ on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\varphi)$ is finite and $n = \max(\mathcal{I}_\pi(\varphi))$, and $\mathcal{C}$ consists of the states $\pi(n)$ on paths $\pi \in \Pi$ where $\mathcal{I}_\pi(\varphi)$ is infinite and $n \in \text{cycle}(\pi)$. Now, let $\mathcal{S} = ((\text{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]}) \cup \mathcal{B}^\gamma_\varphi \cup \mathcal{C}) \cap [\![\gamma[\top]]\!]$ and $\pi \in \Pi$. (i) If $\mathcal{I}_\pi(\varphi)$ is infinite, $n \in \text{cycle}(\pi)$, and $\pi^n \models \gamma[\top]$, we know that $\pi(n) \in \mathcal{S}$ since $\pi(n) \in \mathcal{C} \cap [\![\gamma[\top]]\!]$. (ii) Otherwise, if $\mathcal{I}_\pi(\varphi)$ is infinite, $\pi^i \not\models \gamma[\top]$ for all $i \in \text{cycle}(\pi)$, and n is the highest index in $\mathcal{I}_\pi(\varphi)$ such that $\pi^n \models \gamma[\top]$, we know that $\pi(n) \in \mathcal{S}$ since $\pi(n) \in \text{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]} \cap [\![\gamma[\top]]\!]$. (iii) Otherwise, if $\mathcal{I}_\pi(\varphi)$ is finite, n is the highest index in $\mathcal{I}_\pi(\varphi)$ such that $\pi^n \models \gamma[\top]$, and $n = \max(\mathcal{I}_\pi(\varphi))$, we know that $\pi(n) \in \mathcal{S}$ since $\pi(n) \in \mathcal{B}^\gamma_\varphi \cap [\![\gamma[\top]]\!]$. (iv) Otherwise, if $\mathcal{I}_\pi(\varphi)$ is finite, n is the highest index in $\mathcal{I}_\pi(\varphi)$ such that $\pi^n \models \gamma[\top]$, and $n \neq \max(\mathcal{I}_\pi(\varphi))$, we know that $\pi(n) \in \mathcal{S}$ since $\pi(n) \in \text{pre}_\exists(\mathcal{U}_3) \cap \mathcal{R}_{\varphi \wedge \neg\gamma[\bot]} \cap [\![\gamma[\top]]\!]$. It is easy to see that $\mathcal{S}$ contains no other states. Hence, the assertion follows from item (i) and by combining item (ii), item (iii), and item (iv). □

Now, we are able to prove the correctness of the extended Chan algorithm.

Theorem 5.2. *Let $\gamma \in CTLQ^x$ and $\mathcal{Q}$ be a set of states in a Kripke structure. Moreover, let* EExactSol *be the function defined in Algorithm 2. Suppose that $\mathcal{Q} \models \gamma[\top]$, i.e., γ has a solution at each state in $\mathcal{Q}$. Then,* $\text{EExactSol}(\gamma, \mathcal{Q})$ *returns the unique set of solution states to γ at $\mathcal{Q}$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = ?$. Then, the assertion holds trivially since $\text{ExactSol}(?, \mathcal{Q}) = \mathcal{Q}$ and $\mathcal{Q}$ is the unique set of solution states to ? at $\mathcal{Q}$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{Q})$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{Q}$. Therefore, we have $\mathcal{Q} \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Moreover, by assumption, we have $\mathcal{Q} \models \gamma[\top]$, which implies $\mathcal{Q} \models \theta$. Hence, since $\mathcal{Q} \models \theta \wedge \bar{\gamma}[\varphi]$ iff $\mathcal{Q} \models \theta$ and $\mathcal{Q} \models \bar{\gamma}[\varphi]$, we

know that $\mathcal{Q} \models \theta \wedge \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \theta \vee \bar{\gamma}$. By induction hypothesis, we know that ExactSol($\bar{\gamma}$, $\mathcal{Q} \setminus [\![\theta]\!]$) returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{Q} \setminus [\![\theta]\!]$. Therefore, we have $\mathcal{Q} \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Moreover, note that trivially $\mathcal{Q} \cap [\![\theta]\!] \models \theta$. Hence, since $\mathcal{Q} \models \theta \vee \bar{\gamma}[\varphi]$ iff $\mathcal{Q} \cap [\![\theta]\!] \models \theta$ and $\mathcal{Q} \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$, we know that $\mathcal{Q} \models \theta \vee \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$. By induction hypothesis, we know that ExactSol($\bar{\gamma}$, $\mathrm{post}_\exists(\mathcal{Q})$) returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathrm{post}_\exists(\mathcal{Q})$. Therefore, we have $\mathrm{post}_\exists(\mathcal{Q}) \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Hence, since $\mathcal{Q} \models \mathbf{AX}\,\bar{\gamma}[\varphi]$ iff $\mathrm{post}_\exists(\mathcal{Q}) \models \bar{\gamma}[\varphi]$, we know that $\mathcal{Q} \models \mathbf{AX}\,\bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$. Since $\mathbf{AF}\,\bar{\gamma} \Leftrightarrow \mathbf{A}(\top\,\mathbf{U}\,\bar{\gamma})$, the assertion follows from the case of the strong until operator $\mathbf{U}$.

▷ Let $\gamma = \mathbf{AG}\,\bar{\gamma}$. Since $\mathbf{AG}\,\bar{\gamma} \Leftrightarrow \mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\bot)$ and $\mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\bot) \Leftrightarrow \mathbf{A}((\bot \vee \bar{\gamma})\,\mathring{\mathbf{W}}\,\bot)$, we know that $\mathbf{AG}\,\bar{\gamma} \Leftrightarrow \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{W}}\,\bot)$. Hence, the assertion follows from the case of the overlapping weak until operator $\mathring{\mathbf{W}}$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$. Since $\mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta) \Leftrightarrow \mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\theta) \wedge \mathbf{AF}\,\theta$ and $\mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\theta) \Leftrightarrow \mathbf{A}((\theta \vee \bar{\gamma})\,\mathring{\mathbf{W}}\,\theta)$, we know that $\mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta) \Leftrightarrow \mathbf{A}((\theta \vee \bar{\gamma})\,\mathring{\mathbf{W}}\,\theta) \wedge \mathbf{AF}\,\theta$. Moreover, by assumption, we have $\mathcal{Q} \models \gamma[\top]$, which implies $\mathcal{Q} \models \mathbf{AF}\,\theta$. Hence, the assertion follows from the cases of the overlapping weak until operator $\mathring{\mathbf{W}}$ and disjunction.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta)$. By assumption, we have $\mathcal{Q} \models \gamma[\top]$, which implies $\mathcal{Q} \models \mathbf{AF}\,\theta$. Hence, since $\mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta) \Leftrightarrow \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{W}}\,\theta) \wedge \mathbf{AF}\,\theta$, the assertion follows from the case of the overlapping weak until operator $\mathring{\mathbf{W}}$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma})$ and $\mathcal{F} = \mathrm{FurthestSol}(\bar{\gamma}, \theta, \mathcal{Q}, \mathit{true})$. By induction hypothesis, we know that ExactSol($\bar{\gamma}$, $\mathcal{F}$) returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{F}$. Therefore, we have $\mathcal{F} \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma}[\varphi])$, i.e., for each $\pi \in \mathrm{paths}(\mathcal{Q})$ it holds that $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$. By Lemma 5.1, we know that $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$ iff there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\varphi]$, that is, $\varphi \in \mathrm{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. In particular, if there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\bot]$, then $\pi \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi]$ is trivially true. Hence,

it suffices to consider only paths where $\pi^i \not\models \bar{\gamma}[\bot]$ for all $i \in \mathcal{I}_\pi(\theta)$. Thus, let $\Pi \subseteq \text{paths}(\mathcal{Q})$ be the set of such paths, and for each $\pi \in \Pi$ let us distinguish between two cases:
(i) $\mathcal{I}_\pi(\theta)$ is infinite and $\pi^i \models \bar{\gamma}[\top]$ for some $i \in \text{cycle}(\pi)$: Then, by Proposition 5.1 and Lemma 5.5, we know that $\text{sol}_{\{i\}}(\pi, \bar{\gamma}) = \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$ for every $i \in \text{cycle}(\pi)$ such that $\pi^i \models \bar{\gamma}[\top]$. Now, let $\mathcal{C} = \{i \in \text{cycle}(\pi) \mid \pi^i \models \bar{\gamma}[\top]\}$. Hence, since all states with indices in $\mathcal{C}$ have the same solutions, this implies that $\pi \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi]$ iff $\pi^{\mathcal{C}} \models \bar{\gamma}[\varphi]$.
(ii) $\mathcal{I}_\pi(\theta)$ is finite or $\pi^i \not\models \bar{\gamma}[\top]$ for all $i \in \text{cycle}(\pi)$: Recall that, by assumption, we have $\mathcal{Q} \models \gamma[\top]$, which implies that there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\top]$. In particular, since $\mathcal{I}_\pi(\theta)$ is finite or $\pi^i \not\models \bar{\gamma}[\top]$ for all $i \in \text{cycle}(\pi)$, this implies that there exists a highest index $n \in \mathcal{I}_\pi(\theta)$ such that $\pi^n \models \bar{\gamma}[\top]$. Thus, by Proposition 5.1 and Lemma 5.3, we obtain $\text{sol}_{\{n\}}(\pi, \bar{\gamma}) = \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. Consequently, $\pi \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$.
By Lemma 5.8, we know that $\mathcal{F}$ consists of all states with cycle indices $\mathcal{C}$ according to item (i) on paths $\pi \in \Pi$ satisfying the conditions of item (i), and of all states with highest index $n \in \mathcal{I}_\pi(\theta)$ according to item (ii) on paths $\pi \in \Pi$ satisfying the conditions of item (ii). Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\theta \,\mathbf{U}\, \bar{\gamma}[\varphi])$ iff $\mathcal{F} \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\theta \,\mathbf{U}\, \bar{\gamma}[\varphi])$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

- ▷ Let $\gamma = \mathbf{A}(\theta \,\mathring{\mathbf{U}}\, \bar{\gamma})$. Since $\mathbf{A}(\theta \,\mathring{\mathbf{U}}\, \bar{\gamma}) \Leftrightarrow \mathbf{A}(\theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma}))$, the assertion follows from the cases of the strong until operator $\mathbf{U}$ and conjunction.

- ▷ Let $\gamma = \mathbf{A}(\theta \,\bar{\mathbf{U}}\, \bar{\gamma})$. By assumption, we have $\mathcal{Q} \models \gamma[\top]$, which implies $\mathcal{Q} \models \mathbf{AF}\, \neg\theta$. Hence, since $\mathbf{A}(\theta \,\bar{\mathbf{U}}\, \bar{\gamma}) \Leftrightarrow \mathbf{A}(\theta \,\bar{\mathbf{W}}\, \bar{\gamma}) \wedge \mathbf{AF}\, \neg\theta$, the assertion follows from the case of the disjoint weak until operator $\bar{\mathbf{W}}$.

- ▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathbf{W}\, \theta)$. Since $\mathbf{A}(\bar{\gamma} \,\mathbf{W}\, \theta) \Leftrightarrow \mathbf{A}((\theta \vee \bar{\gamma}) \,\mathring{\mathbf{W}}\, \theta)$, the assertion follows from the cases of the overlapping weak until operator $\mathring{\mathbf{W}}$ and disjunction.

- ▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathring{\mathbf{W}}\, \theta) = \mathbf{A}((\mathbf{G}\, \bar{\gamma}) \vee (\bar{\gamma} \,\mathring{\mathbf{U}}\, \theta))$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta}))$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta})$. Therefore, we have $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta}) \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta)$, i.e., for each $\pi \in \text{paths}(\mathcal{Q})$ it holds that $\pi \models \bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta$. Let us distinguish between two cases:
(i) If $\pi^i \not\models \theta$ for all $i \in \mathbb{N}$, we know that $\pi \models \bar{\gamma}[\varphi] \,\mathring{\mathbf{W}}\, \theta$ iff $\pi \models \mathbf{G}\, \bar{\gamma}[\varphi]$.

(ii) Otherwise, if there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \theta$, it is easy to see that $\pi \models \bar{\gamma}[\varphi]\,\mathring{\mathbf{W}}\,\theta$ iff $\pi^i \models \bar{\gamma}[\varphi]$ for all $i \leq n$.
By Lemma 5.6, we know that $\mathcal{R}_{\neg\theta}$ consists of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ such that $\pi^i \not\models \theta$ for all $i \leq n$. Hence, $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta})$ consists of all states $\pi(n)$ on paths $\pi \in \text{paths}(\mathcal{Q})$ such that $\pi^i \not\models \theta$ for all $i < n$. Thus, it is easy to see that $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta})$ consists of all states on paths $\pi \in \text{paths}(\mathcal{Q})$ satisfying the condition of item (i), and all states $\pi(i)$ with $i \leq n$, where n is the least number such that $\pi^n \models \theta$, on paths $\pi \in \text{paths}(\mathcal{Q})$ satisfying the condition of item (ii). Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\bar{\gamma}[\varphi]\,\mathring{\mathbf{W}}\,\theta)$ iff $\mathcal{Q} \cup \text{post}_\exists(\mathcal{R}_{\neg\theta}) \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\bar{\gamma}[\varphi]\,\mathring{\mathbf{W}}\,\theta)$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,\bar{\gamma}))$ and $\mathcal{F} = \text{FurthestSol}(\bar{\gamma}, \theta, \mathcal{Q}, \mathit{false})$. By induction hypothesis, we know that $\text{ExactSol}(\bar{\gamma}, \mathcal{F})$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $\mathcal{F}$. Therefore, we have $\mathcal{F} \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}[\varphi])$, i.e., for each $\pi \in \text{paths}(\mathcal{Q})$ it holds that $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$.
Obviously, if $\pi \not\models \mathbf{G}\,\theta$, it holds that $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$ iff there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\varphi]$, that is, $\varphi \in \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. In particular, if there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\bot]$, then $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$ is trivially true. Also in the case of $\pi \models \mathbf{G}\,\theta$, we trivially obtain $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$. Hence, since $\pi \not\models \mathbf{G}\,\theta$ iff $\mathcal{I}_\pi(\theta)$ is finite, it suffices to consider only paths where $\mathcal{I}_\pi(\theta)$ is finite and $\pi^i \not\models \bar{\gamma}[\bot]$ for all $i \in \mathcal{I}_\pi(\theta)$. Thus, let $\Pi \subseteq \text{paths}(\mathcal{Q})$ be the set of such paths and $\pi \in \Pi$.
Recall that by assumption $\mathcal{Q} \models \gamma[\top]$, which implies that there exists $i \in \mathcal{I}_\pi(\theta)$ such that $\pi^i \models \bar{\gamma}[\top]$. In particular, since $\mathcal{I}_\pi(\theta)$ is finite, this implies that there exists a highest index $n \in \mathcal{I}_\pi(\theta)$ such that $\pi^n \models \bar{\gamma}[\top]$. Thus, by Proposition 5.1 and Lemma 5.3, we obtain $\text{sol}_{\{n\}}(\pi, \bar{\gamma}) = \text{sol}_{\mathcal{I}_\pi(\theta)}(\pi, \bar{\gamma})$. Consequently, $\pi \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$.
By Lemma 5.7, we know that $\mathcal{F}$ consists of all states with highest index $n \in \mathcal{I}_\pi(\theta)$ on paths $\pi \in \Pi$ such that $\pi^n \models \bar{\gamma}[\top]$. Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}[\varphi])$ iff $\mathcal{F} \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}[\varphi])$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{W}}\,\bar{\gamma})$. Since $\mathbf{A}(\theta\,\mathring{\mathbf{W}}\,\bar{\gamma}) \Leftrightarrow \mathbf{A}(\theta\,\mathbf{W}\,(\theta \wedge \bar{\gamma}))$, the assertion follows from the cases of the weak until operator $\mathbf{W}$ and conjunction.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\bar{\mathbf{U}}\,\bar{\gamma}))$. By induction hypothesis, we know that $\mathrm{ExactSol}(\bar{\gamma}, (\mathcal{Q} \cup \mathrm{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!])$ returns the set of solution states $\mathcal{S}$ to $\bar{\gamma}$ at $(\mathcal{Q} \cup \mathrm{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!]$. Therefore, we have $(\mathcal{Q} \cup \mathrm{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$ iff $\mathcal{S} \models \varphi$. Now, consider $\mathcal{Q} \models \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi])$, i.e., for each $\pi \in \mathrm{paths}(\mathcal{Q})$ it holds that $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi]$.
If $\pi \models \mathbf{G}\,\theta$, then $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi]$ is trivially true. Hence, it suffices to consider only paths where $\pi \not\models \mathbf{G}\,\theta$. Thus, let $\Pi \subseteq \mathrm{paths}(\mathcal{Q})$ be the set of such paths and $\pi \in \Pi$. Then, there exists a least $n \in \mathbb{N}$ such that $\pi^n \not\models \theta$. It is easy to see that $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi]$ iff $\pi^n \models \bar{\gamma}[\varphi]$.
By Lemma 5.6, we know that $\mathcal{R}_\theta$ consists of all states $\pi(n)$ on paths $\pi \in \mathrm{paths}(\mathcal{Q})$ such that $\pi^i \models \theta$ for all $i \leq n$. Hence, $\mathcal{Q} \cup \mathrm{post}_\exists(\mathcal{R}_\theta)$ consists of all states $\pi(n)$ on paths $\pi \in \mathrm{paths}(\mathcal{Q})$ such that $\pi^i \models \theta$ for all $i < n$. Thus, it is easy to see that $(\mathcal{Q} \cup \mathrm{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!]$ consists of all states $\pi(n)$ on paths $\pi \in \Pi$, where n is the least number such that $\pi^n \not\models \theta$. Therefore, it holds that $\mathcal{Q} \models \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi])$ iff $(\mathcal{Q} \cup \mathrm{post}_\exists(\mathcal{R}_\theta)) \setminus [\![\theta]\!] \models \bar{\gamma}[\varphi]$. Hence, $\mathcal{Q} \models \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi])$ iff $\mathcal{S} \models \varphi$, i.e., $\mathcal{S}$ is the set of solution states to γ at $\mathcal{Q}$.

This concludes the proof. □

Remark 5.4. Similar to the original Chan algorithm, the extended Chan algorithm can also be improved in several ways (cf. Remark 5.3). Moreover, note that the assumption in Theorem 5.2 that γ has a solution, can be simply checked by a single model checking call before applying the extended Chan algorithm. Note that this assumption is much weaker than the requirement of validity in the original Chan algorithm. In particular, when solving a query is reduced to solving a subquery at several states, from validity it follows that the subquery has a solution at each of these states. From the assumption in Theorem 5.2, however, it follows that there is at least one state at which the subquery has a solution. This fact is also reflected in the proof of Theorem 5.2.

As already mentioned in the case of the original Chan algorithm, in order to obtain an exact solution, we have to compute the strongest formula (within the set of interesting solutions) that holds at the unique set of solution states. In particular, the following corollary shows how to obtain a propositional exact solution by the extended Chan algorithm. Its proof is analogous to the proof of Corollary 5.1.

Corollary 5.2. *Let $\gamma \in CTLQ^x$ and s_0 be the initial state of a Kripke structure $\mathfrak{K}$. Moreover, let* EExactSol *be the function defined in Algorithm 2. Then, the characteristic function of the set returned by* EExactSol$(\gamma, \{s_0\})$ *is a propositional exact solution to γ in $\mathfrak{K}$.*

5.2.4 Non-propositional Solutions

By Corollary 5.1 and Corollary 5.2, we have already shown how to obtain propositional exact solutions from the results of the Chan and the extended Chan algorithm respectively. In this section, we investigate the computation of non-propositional solutions based on the set of solution states to a given query. Of course, by the definition of a set of solution states, every formula that holds at such a set is a solution. However, we are interested in a systematic computation of such formulas that are as strong as possible. To this aim, we show how to compute non-propositional solutions by modifying the functions ExactSol and EExactSol defined in Algorithm 1 and Algorithm 2 respectively.

Since we are able to compute propositional exact solutions to queries in CTLQ^x, the idea behind our approach is to solve queries $\gamma' \in$ CTLQ^x at the set of solution states to a given query $\gamma \in$ CTLQ^x. Then, if φ is a propositional exact solution to γ', the formula $\gamma'[\varphi]$ must be a non-propositional solution to γ. Obviously, the main restriction of this approach is that γ' must be in CTLQ^x such that the extended Chan algorithm can be applied, i.e., we are not able to compute all solutions in this way.

Example 5.5. For example, if we are interested in solutions that consist of propositional formulas and nested next operators $\mathbf{AX}$, the queries γ' to be solved are of the form ?, $\mathbf{AX}$?, $\mathbf{AX}\,\mathbf{AX}$?, $\mathbf{AX}\,\mathbf{AX}\,\mathbf{AX}$?, ... Thus, if φ_0, φ_1, φ_2, φ_3, ... are the corresponding propositional exact solutions, we know that φ_0, $\mathbf{AX}\,\varphi_1$, $\mathbf{AX}\,\mathbf{AX}\,\varphi_2$, $\mathbf{AX}\,\mathbf{AX}\,\mathbf{AX}\,\varphi_3$, ... are non-propositional solutions to the original query $\gamma \in$ CTLQ^x. Since γ is distributive over conjunction, we know that also $\varphi_0 \wedge \mathbf{AX}(\varphi_1 \wedge \mathbf{AX}(\varphi_2 \wedge \mathbf{AX}(\varphi_3 \wedge \cdots)))$ is a solution to γ. In fact, this solution must be relatively strong, since the propositional solutions in it are exact.

We will now show how such solutions as in the above example can be systematically computed by modifying the Chan resp. extended Chan algorithm. To this aim, the maximum nesting depth n of the next operators

is added as third argument to the functions ExactSol and EExactSol. This number remains unchanged for all recursive calls except in the case of the placeholder (i.e., Line 3 in Algorithm 1 resp. Line 11 in Algorithm 2), which is redefined in the following way (for the case of Algorithm 1):

if $n = 0$ **then**
  return $\mathrm{char}_{\mathcal{P}}(\mathcal{Q})$;
else
  return $\mathrm{char}_{\mathcal{P}}(\mathcal{Q}) \wedge \mathbf{AX}\ \mathrm{ExactSol}(\mathbf{AX}?, \mathcal{Q}, n-1)$;

If $n = 0$, then the modified algorithm returns in principle the same result as before. In particular, it returns the characteristic function $\mathrm{char}_{\mathcal{P}}$ restricted to a set of interesting atomic propositions $\mathcal{P}$ instead of the set of solution states. Otherwise, if $n > 0$, it returns a solution consisting of propositional exact solutions and nested next operators of the form

$$\varphi_0 \wedge \mathbf{AX}(\varphi_1 \wedge \mathbf{AX}(\varphi_2 \wedge \cdots \mathbf{AX}(\varphi_{n-1} \wedge \mathbf{AX}\,\varphi_n)\cdots)),$$

where φ_i are propositional exact solutions for all $0 \leq i \leq n$. The correctness of this modification can be easily proved by induction on n. In the same way, several other kinds of non-propositional solutions with fixed nesting depth of the temporal operators occurring in them can be computed as long as the Chan resp. extended Chan algorithm is applicable.

5.3 Further Approaches

After the publication of the first algorithm for solving temporal logic queries, namely the Chan algorithm [Cha00], there has been active research on more general query solving algorithms. All these approaches were developed in order to compute the set of propositional minimal solutions. In this section, we summarize their basic ideas and we show how they can be extended in order to compute also non-propositional solutions.

This will be done in analogy to the extension of the Chan algorithm in Section 5.2.4. In particular, our focus lies on the case of solutions with bounded nesting depth of temporal operators, where the next operator is the only allowed temporal operator. It is then easy to see how extensions for arbitrary non-propositional solutions can be obtained.

The common basic principle of the following query solving algorithms is to evaluate a query recursively on its syntactic structure. Thus, in order to obtain non-propositional solutions, it suffices to redefine the case of the placeholder (cf. Section 5.2.4). To this aim, let us first introduce some notations. Since the solutions we are interested in consist of nested **AX** and **EX** operators, we define $\gamma_{\mathrm{A}} = \mathbf{AX}?$ and $\gamma_{\mathrm{E}} = \mathbf{EX}?$. Moreover, for every query γ and every set of formulas Φ, we define $\gamma \diamond \Phi = \bigwedge\{\gamma[\varphi] \mid \varphi \in \Phi\}$. For example, if $\gamma = \mathbf{AX}\,\mathbf{EX}?$ and $\Phi = \{p \wedge \neg q, \mathbf{AF}\,q, p \vee \mathbf{EX}\,p\}$, then $\gamma \diamond \Phi = \mathbf{AX}\,\mathbf{EX}\,(p \wedge \neg q) \wedge \mathbf{AX}\,\mathbf{EX}\,\mathbf{AF}\,q \wedge \mathbf{AX}\,\mathbf{EX}(p \vee \mathbf{EX}\,p)$.

5.3.1 Extended Alternating Automata

Bruns and Godefroid [BG01] showed how to solve queries of any temporal logic having a translation to alternating automata. To this aim, they adapted the automata-theoretic approach of Kupferman et al. [KVW00]. The key idea of this adaption is to generalize the transition function of alternating automata in such a way that disjunction and conjunction are replaced by special *meet* and *join* operations of an arbitrary finite lattice over sets of propositional formulas:

$$\mathcal{A} \underline{\wedge} \mathcal{B} = \min_{\Rightarrow}(\{a \vee b \mid a \in \mathcal{A}, b \in \mathcal{B}\}) \quad \text{and} \quad \mathcal{A} \underline{\vee} \mathcal{B} = \min_{\Rightarrow}(\mathcal{A} \cup \mathcal{B})$$

The resulting automata are called *extended alternating automata (EAA)*. To solve a query γ in a Kripke structure $\mathfrak{K}$, the query has to be translated into an EAA $\mathfrak{A}_\gamma$ and the product automaton of $\mathfrak{A}_\gamma$ and $\mathfrak{K}$ has to be built. Each node of an accepting run of the resulting product automaton is labeled with an element of the underlying lattice (i.e., a set of propositional formulas). The maximum value labeling the root of an accepting run of the product automaton is the set of minimal solutions to γ in $\mathfrak{K}$. This maximum value is computed by simultaneously checking non-emptiness for every value of the underlying lattice.

Obviously, this approach can be generalized to arbitrary finite lattices even over sets of non-propositional formulas. In particular, in order to guarantee the finiteness of the lattice, we add the maximum nesting depth n of temporal operators as fourth argument to the transition function. Note that we restrict our considerations to the temporal operators used in [BG01]. Moreover, note that we use γ_1 and γ_2 to denote both queries and formulas.

Then, for all sets $\mathcal{P} \subseteq \mathcal{A}$ of atomic propositions and arities $k \in \mathcal{D} \subset \mathbb{N}$, the adapted transition function ρ is given by:

$$
\begin{aligned}
\rho(p, \mathcal{P}, k, n) &= \begin{cases} \{\bot\} & : \ p \in \mathcal{P} \\ \emptyset & : \ p \in \mathcal{A} \setminus \mathcal{P} \end{cases} \\
\rho(\neg p, \mathcal{P}, k, n) &= \begin{cases} \emptyset & : \ p \in \mathcal{P} \\ \{\bot\} & : \ p \in \mathcal{A} \setminus \mathcal{P} \end{cases} \\
\rho(\gamma_1 \wedge \gamma_2, \mathcal{P}, k, n) &= \rho(\gamma_1, \mathcal{P}, k, n) \mathbin{\underline{\wedge}} \rho(\gamma_2, \mathcal{P}, k, n) \\
\rho(\gamma_1 \vee \gamma_2, \mathcal{P}, k, n) &= \rho(\gamma_1, \mathcal{P}, k, n) \mathbin{\underline{\vee}} \rho(\gamma_2, \mathcal{P}, k, n) \\
\rho(\mathbf{AX}\,\gamma, \mathcal{P}, k, n) &= \underline{\bigwedge}_{c=1}^{k} (c, \gamma, n) \\
\rho(\mathbf{EX}\,\gamma, \mathcal{P}, k, n) &= \underline{\bigvee}_{c=1}^{k} (c, \gamma, n) \\
\rho(\mathbf{A}(\gamma_1 \,\mathbf{U}\, \gamma_2), \mathcal{P}, k, n) &= \\
&\quad \rho(\gamma_2, \mathcal{P}, k, n) \mathbin{\underline{\vee}} (\rho(\gamma_1, \mathcal{P}, k, n) \mathbin{\underline{\wedge}} \underline{\bigwedge}_{c=1}^{k} (c, \mathbf{A}(\gamma_1 \,\mathbf{U}\, \gamma_2), n)) \\
\rho(\mathbf{E}(\gamma_1 \,\mathbf{U}\, \gamma_2), \mathcal{P}, k, n) &= \\
&\quad \rho(\gamma_2, \mathcal{P}, k, n) \mathbin{\underline{\vee}} (\rho(\gamma_1, \mathcal{P}, k, n) \mathbin{\underline{\wedge}} \underline{\bigvee}_{c=1}^{k} (c, \mathbf{E}(\gamma_1 \,\mathbf{U}\, \gamma_2), n)) \\
\rho(\mathbf{A}(\gamma_1 \,\mathbf{R}\, \gamma_2), \mathcal{P}, k, n) &= \\
&\quad \rho(\gamma_2, \mathcal{P}, k, n) \mathbin{\underline{\wedge}} (\rho(\gamma_1, \mathcal{P}, k, n) \mathbin{\underline{\vee}} \underline{\bigwedge}_{c=1}^{k} (c, \mathbf{A}(\gamma_1 \,\mathbf{R}\, \gamma_2), n)) \\
\rho(\mathbf{E}(\gamma_1 \,\mathbf{R}\, \gamma_2), \mathcal{P}, k, n) &= \\
&\quad \rho(\gamma_2, \mathcal{P}, k, n) \mathbin{\underline{\wedge}} (\rho(\gamma_1, \mathcal{P}, k, n) \mathbin{\underline{\vee}} \underline{\bigvee}_{c=1}^{k} (c, \mathbf{E}(\gamma_1 \,\mathbf{R}\, \gamma_2), n))
\end{aligned}
$$

Intuitively, the arguments of the transition function ρ are the query γ, the set of atomic propositions $\mathcal{P}$ labeling the actual state, the number of successors k of the actual state, and the nesting depth n. The value of ρ is the set of minimal solutions to γ at the actual state.

The most interesting case of this generalization, however, is the case of the placeholder, which is redefined in the following way:

$$
\rho(?, \mathcal{P}, k, n) = \begin{cases} \{\bigwedge\{p \mid p \in \mathcal{P}\} \wedge \bigwedge\{\neg p \mid p \in \mathcal{A} \setminus \mathcal{P}\}\} & : \ n = 0 \\ \left.\begin{array}{l} \rho(?, \mathcal{P}, k, 0) \mathbin{\underline{\vee}} \\ \{\gamma_{\mathrm{A}} \diamond \rho(\gamma_{\mathrm{A}}, \mathcal{P}, k, n-1)\} \mathbin{\underline{\vee}} \\ \{\gamma_{\mathrm{E}} \diamond \rho(\gamma_{\mathrm{E}}, \mathcal{P}, k, n-1)\} \end{array}\right\} & : \ n > 0 \end{cases}
$$

Note that this is the only case where the nesting depth n of temporal operators of the solutions is changed. If $n = 0$, then the returned solution is purely propositional. Otherwise, if $n > 0$, the returned solutions consist of a combination of propositional solutions and universal and existential quantified next operators. The minimality of these solutions is guaranteed by the join operator $\underline{\vee}$. Because of the stepwise recursive construction, non-minimal solutions are excluded as early as possible. It is easy to see that also other non-propositional solutions can be computed in this way.

5.3.2 Multi-valued Model Checking

Chechik, Devereux, and Gurfinkel [GDC02, GCD03, CG03, GCD03] investigated CTL query solving by using their multi-valued model checker χChek. Multi-valued CTL model checking is based on two extensions of standard model checking: (i) The labeling function of Kripke structures allows variables not only to be true or false at a particular state but to have a value of an underlying lattice. (ii) The temporal logic CTL is extended to χCTL in order to refer to such values within formulas.

In the case of query solving, they used a lattice over sets of propositional formulas for their multi-valued model checking framework. Query solving is then reduced to multi-valued model checking by translating a given query into a χCTL formula such that the value of this formula in the model is the set of minimal solutions to the query.

In analogy to Bruns and Godefroid (cf. Section 5.3.1), this approach can also be generalized to arbitrary finite lattices even over sets of non-propositional formulas. In particular, in order to guarantee the finiteness of the lattice, we add the maximum nesting depth n of temporal operators as argument to the definition of the multi-valued semantics. Note that we restrict our considerations to the temporal operators used in [GDC02]. Moreover, note that we use γ_1 and γ_2 to denote both queries and formulas. Let $\neg$ (negation), $\sqcap$ (meet), and $\sqcup$ (join) be the usual lattice operations. Then, for any lattice $(\mathcal{L}, \Rightarrow)$ and interpretation function $\mathcal{I} : \mathcal{Q} \times \mathcal{A} \rightarrow \mathcal{L}$ assigning to each pair of an atomic proposition and a state an element of the underlying lattice, the adapted multi-valued semantics is given by:

$$\begin{aligned} [\![\varphi]\!]_n(s) &= \varphi, \text{ for } \varphi \in \mathcal{L} \\ [\![p]\!]_n(s) &= \mathcal{I}(s,p), \text{ for } p \in \mathcal{A} \end{aligned}$$

$$
\begin{aligned}
[\![\neg\gamma]\!]_n(s) &= \neg[\![\gamma]\!]_n(s) \\
[\![\gamma_1 \vee \gamma_2]\!]_n(s) &= [\![\gamma_1]\!]_n(s) \sqcup [\![\gamma_2]\!]_n(s) \\
[\![\mathbf{EX}\,\gamma]\!]_n(s) &= \bigsqcup_{(s,s')\in\Delta} [\![\gamma]\!]_n(s') \\
[\![\mathbf{EG}\,\gamma]\!]_n(s) &= \bigsqcup_{\pi\in\mathrm{paths}(s)} \bigsqcap_{i\in\mathbb{N}} [\![\gamma]\!]_n(\pi(i)) \\
[\![\mathbf{E}(\gamma_1\,\mathbf{U}\,\gamma_2)]\!]_n(s) &= \bigsqcup_{\pi\in\mathrm{paths}(s)} \bigsqcup_{j\in\mathbb{N}} ([\![\gamma_2]\!]_n(\pi(j)) \sqcap \bigsqcap_{i<j} [\![\gamma_1]\!]_n(\pi(i)))
\end{aligned}
$$

The most interesting case of this generalization, however, is again the case of the placeholder, which is redefined in the following way by using the set of all characteristic functions over the set of atomic propositions $\mathcal{A}$ defined by $\Phi = \{\bigwedge\{p \mid p \in \mathcal{P}\} \wedge \bigwedge\{\neg p \mid p \in \mathcal{A} \setminus \mathcal{P}\} \mid \mathcal{P} \subseteq \mathcal{A}\}$:

$$
[\![?]\!]_n(s) = \begin{cases} \bigsqcup_{\varphi\in\Phi}([\![\varphi]\!]_0(s) \sqcap \uparrow\{\varphi\}) & : \; n = 0 \\ [\![?]\!]_0(s) \sqcup \{\gamma_\mathrm{A} \diamond [\![\gamma_\mathrm{A}]\!]_{n-1}(s)\} \sqcup \{\gamma_\mathrm{E} \diamond [\![\gamma_\mathrm{E}]\!]_{n-1}(s)\} & : \; n > 0 \end{cases}
$$

Note that this is the only case where the nesting depth n of temporal operators of the solutions is changed. If $n = 0$, then the returned solution is purely propositional, namely the expression $\uparrow\{\varphi\} = \{\varphi' \mid \varphi \Rightarrow \varphi'\}$ where φ is the characteristic function of the actual state. Otherwise, if $n > 0$, the returned solutions consist of a combination of propositional solutions and universal and existential quantified next operators. The minimality of these solutions is guaranteed by the join operator $\sqcup$. Because of the stepwise recursive construction, non-minimal solutions are excluded as early as possible. Again, it is easy to see that also other non-propositional solutions can be computed in this way.

5.3.3 Reductions to Model Checking

Hornus and Schnoebelen [HS02] dealt with more theoretical results on query solving for arbitrary fragments of CTL*. They showed that deciding whether there exists a single minimal solution in a *fixed* model and computing this solution can be reduced to a linear number (in the size of the model) of model checking calls. Moreover, they showed that a second

minimal solution can be reduced to a quadratic number, a third minimal solution to a cubic number, etc. of model checking calls. Concerning the number of minimal solutions to CTL queries, they proved that deciding whether there exist at least k (in unary) minimal solutions is NP-complete and counting the number of minimal solutions is $\sharp$P-complete.

Although they did not present an explicit algorithm, the following steps can be extracted from their proofs. In order to compute the set of propositional minimal solutions to a query γ, a propositional formula φ is defined and it is checked whether $\gamma[\varphi]$ holds in the model. Then, φ is modified in a sophisticated way and $\gamma[\varphi]$ is checked again. This procedure is repeated until φ is identified to be a new minimal solution.

It is easy to see that also non-propositional solutions to a query γ can be computed in this way. To this aim, compute the set Φ of propositional minimal solutions to the query $\gamma[\gamma']$, where γ' is any query. Then, $\gamma' \diamond \Phi$ is obviously a non-propositional solution to γ. However, since Hornus and Schnoebelen did not present a recursive algorithm over the syntactic structure of queries, there is no natural extension for computing minimal non-propositional solutions as in the approaches described above.

5.4 Summary

Temporal operators can be distinguished into *universal* and *existential* ones with respect to each of their operands. For example, the global operator $\mathbf{G}$ is universal, since evaluating $\pi \models \mathbf{G}\,\varphi$ can be reduced to evaluating $\forall i \in \mathbb{N}.\ \pi^i \models \varphi$. Contrary, the future operator $\mathbf{F}$ is existential, since evaluating $\pi \models \mathbf{F}\,\varphi$ can be reduced to evaluating $\exists i \in \mathbb{N}.\ \pi^i \models \varphi$.

In general, existential operators cause a high computational effort when solving temporal logic queries. For example, in order to compute all solutions to $\mathbf{F}\,\gamma$ on path π, the subquery γ has to be solved at all positions on π. However, if γ is *intermediate collecting* (as defined in Chapter 4), solving $\mathbf{F}\,\gamma$ can be reduced to solving γ at a single position on π. In this way, it is possible to eliminate non-determinism in the form of existential choices when solving queries. Intuitively, this is done by computing a distance depending on the temporal operator and choosing states that are furthest away from the actual state but not further away than the computed distance and such that the subquery has a solution at the chosen states. Although

this condition seems to be quite complex, it can be efficiently implemented by symbolic algorithms as it was done by Chan.

Chan's symbolic algorithm for solving temporal logic queries in the syntactic fragment CTLQ^x is based on the kind of determinization described above. In particular, all immediate subqueries of existential operators in CTLQ^x are intermediate collecting. Hence, this property can be exploited in order to solve such queries more efficiently. In fact, Chan's original algorithm requires also another property, namely the existence of a solution in every Kripke structure. However, the original algorithm can be extended in order to solve all queries in CTLQ^x. The main contributions of this chapter were the presentation of such an extension and the correctness proofs of both the Chan and the extended Chan algorithm.

Chapter 6

Parameterized Vacuity

6.1 Introduction

When a model checker detects that a specification φ is violated, it will output a counterexample. If the specification is satisfied, however, there is usually no feedback from the model checker; in particular, the user does not know whether φ is satisfied *vacuously*, i.e., due to a trivial reason. One of the simplest examples of vacuous satisfaction is antecedent failure [BB94], i.e., the situation when the antecedent φ of an implication $\varphi \Rightarrow \psi$ is not satisfied in the model, resulting in the vacuous truth of $\varphi \Rightarrow \psi$. Since experience has shown that vacuous satisfaction often hints at an error either in the model or in the specification, vacuity detection has gained much interest in the last years from both industry and academia. To cite Beer et al. from the IBM Haifa Research Laboratory [BBDER97]:

> *Both the ability to detect trivial passes and the ability to generate interesting witnesses are of great importance in the practical application of formal verification to hardware design. Our experience has shown that typically 20% of formulas pass vacuously during the first formal verification runs of a new hardware design, and that vacuous passes always point to a real problem in either the design or its specification or environment. Of the formulas which pass non-vacuously, examination of the witness traces discovers a problem for approximately 10% of the formulas.*

Intuitively, vacuity means that the truth value of a formula φ is independent of the truth value of a subformula, i.e., the subformula can be replaced by any other formula without changing the truth value of φ. Thus,

a naive approach for detecting vacuity would be to check the truth value of a formula for all possible substitutions of all subformulas. However, this is obviously infeasible in practice. Therefore, Kupferman and Vardi showed in their seminal paper [KV99], that in the case of CTL*, vacuity detection can be reduced to model checking if the subformulas occur with pure polarity (i.e., under an even or an odd number of negations, but not mixed). In particular, they showed that a formula φ is vacuously satisfied with respect to a subformula ψ iff the truth value of φ remains unchanged when replacing ψ by the constant truth values $\top$ (*true*) resp. $\bot$ (*false*) depending on the polarity of ψ in φ. Thereafter, Beer et al. [BBDER01] generalized this result to any logic with polarity.

Example 6.1 *(Classical vacuity).* Note that the specification $\mathbf{AX}(p \vee \mathbf{AX}\, q)$ is trivially satisfied in every model where the stronger formula $\mathbf{AX}\, p$ holds (since $\mathbf{AX}\, p \Rightarrow \mathbf{AX}(p \vee \mathbf{AX}\, q)$). In this case, the subformula $\mathbf{AX}\, q$ can be replaced by $\bot$, that is, $\mathbf{AX}(p \vee \bot) = \mathbf{AX}\, p$, without affecting the truth value. Hence, $\mathbf{AX}(p \vee \mathbf{AX}\, q)$ is vacuously satisfied.

The main motivation for the work presented in this chapter is the observation (already mentioned by Beer et al. [BBDER01]) that the common notion of vacuity described above does not suffice to capture the intuitive range of "trivial" satisfaction.

Example 6.2 *(Limits of classical vacuity).* Note that the specification $\mathbf{AX}\, \mathbf{AF}\, p$ is trivially satisfied in every model where the stronger formula $\mathbf{AX}\, p$ holds (since $\mathbf{AX}\, p \Rightarrow \mathbf{AX}\, \mathbf{AF}\, p$). This form of trivial satisfaction however, does not fall under the common notion of vacuity since neither p nor $\mathbf{AF}\, p$ can be replaced by $\bot$ without affecting the truth value. A similar example due to Pnueli [Pnu97] will be described later in more detail.

Therefore, we suggest a refined notion of vacuity (*weak vacuity*) which is parameterized by a user-defined class Θ of *vacuity causes*. Under this notion, a specification is vacuously satisfied if a subformula collapses to a vacuity cause, and classical vacuity amounts to the special case where Θ consists of the constant truth values $\top$ and $\bot$.

Example 6.3. Let $\Theta = \{\top, \bot, p\}$ be a given set of vacuity causes. Then, the subformula $\mathbf{AX}\, q$ in Example 6.1 can be replaced by $\bot \in \Theta$ without

affecting the truth value. Hence, the specification is vacuously satisfied in the classical sense. Moreover, in Example 6.2, the subformula $\mathbf{AF}\,p$ can be replaced by the stronger formula $p \in \Theta$ without affecting the truth value. Hence, the specification is vacuously satisfied in our weaker sense.

In addition to a generalization of classical vacuity, we establish a close relationship between the detection of weak vacuity and temporal logic query solving, which gives rise to a systematic framework for weak vacuity.

This chapter is organized as follows: In Section 6.2, we summarize the basic knowledge of vacuity detection that we need in this chapter. Then, we present our generalization of vacuity in Section 6.3 consisting of three subsections. Section 6.3.1 shows how the classical vacuity notion can be embedded into temporal logic queries. The most important part of this chapter is then presented in Section 6.3.2, where we introduce our generalization to *weak vacuity*. Afterwards, Section 6.3.3 shows how to construct vacuity witnesses for weak vacuity. In Section 6.4, we give a short overview of related work. Finally, we summarize in Section 6.5.

6.2 Background on Vacuity Detection

Since most of the definitions and results concerning vacuity use the substitution of subformulas of a given formula, Kupferman and Vardi [KV99] introduced the notation $\varphi[\psi \leftarrow \theta]$ to denote the result of substituting the subformula ψ (i.e., all occurrences of ψ) of φ by θ. This enables us to formulate the following two definitions which are adapted from [BBDER97].

Definition 6.1 (Affect). The subformula ψ of formula φ *affects* φ in a model $\mathfrak{M}$ iff there is a formula θ such that the truth values of φ and $\varphi[\psi \leftarrow \theta]$ are different in $\mathfrak{M}$.

With this notion at hand, we are able to define what we mean by vacuity.

Definition 6.2 (Vacuity). The model $\mathfrak{M}$ satisfies φ *vacuously* iff $\mathfrak{M} \models \varphi$ and there is some subformula ψ of φ such that ψ does not affect φ in $\mathfrak{M}$.

Note that according to this definition, all subformulas have to be checked in order to detect non-vacuity. For practical reasons, this can be easily

modified in such a way that vacuity is checked with respect to user-selected subformulas that are considered to be relevant. Therefore, a special kind of vacuity was introduced by Beer et al. [BBDER01], namely ψ-vacuity. We slightly adapt their definition in order to preserve a consistent terminology.

Definition 6.3 (ψ-Vacuity). Let ψ be a subformula of formula φ. The model $\mathfrak{M}$ satisfies φ *ψ-vacuously* iff $\mathfrak{M} \models \varphi$ and ψ does not affect φ in $\mathfrak{M}$.

It is easy to see that a formula is vacuously satisfied iff it is ψ-vacuously satisfied for some subformula ψ. Kupferman and Vardi [KV99] showed that if there exist multiple occurrences of ψ in φ, then vacuity detection is much harder because one occurrence of the subformula may be positive (i.e., under an even number of negations) and another occurrence of the same subformula may be negative (i.e., under an odd number of negations). Therefore, they required that every subformula occurs only once, which guarantees that the substituted subformula occurs with pure polarity (i.e., either positive or negative) which, on the other hand, guarantees some kind of monotonicity. Under this assumption, they showed that checking ψ-vacuity of a satisfied formula φ can be reduced to model checking $\varphi[\psi \leftarrow \bot]$ (if ψ occurs positive) resp. $\varphi[\psi \leftarrow \top]$ (if ψ occurs negative).

6.3 From Vacuity to Parameterized Vacuity

In this section, we show that temporal logic queries can be seen as a uniform framework for vacuity detection and how the conventional concept of vacuity can be nicely generalized by using terms of temporal logic queries. To this aim, consider the query $\gamma = \varphi[\psi \leftarrow ?]$, which we obtain by replacing subformula ψ by the placeholder. Obviously, it holds that $\gamma[\theta] = \varphi[\psi \leftarrow \theta]$, which indicates how to use temporal logic queries for vacuity detection.

Definition 6.4 (Annotate). A query γ *annotates* a formula φ iff it holds that $\gamma[\psi] = \varphi$ for some subformula ψ of φ.

This definition enables us to encode selected occurrences of a subformula ψ in φ into a query γ. Then, checking if ψ affects φ can be done by determining the solutions to γ. Hence, checking vacuity can be reduced to query solving. In order to do this, we need another definition.

Definition 6.5 (Equivalent). A query γ is *equivalent* to a formula φ in a model $\mathfrak{M}$, in symbols $\gamma \equiv_{\mathfrak{M}} \varphi$, iff for all formulas θ, $\gamma[\theta] \Leftrightarrow \varphi$ in $\mathfrak{M}$.

6.3.1 Strong Vacuity

Now, we are able to define the classical notion of vacuity by terms of temporal logic queries. We call it *strong* vacuity because later we will also define a more general and therefore weaker form of vacuity.

Definition 6.6 (Strong vacuity). Let φ be a formula annotated by γ. Then, the model $\mathfrak{M}$ satisfies φ *strong* γ*-vacuously* iff $\mathfrak{M} \models \varphi$ and $\gamma \equiv_{\mathfrak{M}} \varphi$.

Obviously, a comparison between vacuity and strong vacuity makes only sense when the set of annotating queries that are taken into account for strong vacuity detection contains only those queries in which all occurrences of a subformula are simultaneously replaced by the placeholder. More formally, the annotating queries must be given by the set

$$\{\varphi[\psi \leftarrow ?] \mid \psi \text{ is some subformula of } \varphi\}.$$

Then, it is easy to see that classical vacuity and strong vacuity coincide. This is not surprising because so far we have essentially reformulated the notion of vacuity by terms of temporal logic queries. However, as we will see later, temporal logic queries provide us with another point of view of vacuity which will be crucial towards a generalization.

Recall that Kupferman and Vardi [KV99] investigated vacuity of CTL* formulas with respect to subformulas with pure polarity (i.e., either positive or negative). Beer et al. [BBDER01] showed that this approach can be generalized to any logic with polarity. In terms of temporal logic queries, this can be further generalized by considering arbitrary monotonic queries. Note that pure polarity implies monotonicity but not vice versa.

The following lemma is our first step towards a generalization. Its proof is similar to the proof of Theorem 1 in [KV99]. The main difference is the use of Lemma 2.1, which is based on the purely semantical property of monotonicity in contrast to Lemma 1 in [KV99], which is based on the syntactic (and therefore more specific) property of pure polarity. Note that it suffices to consider only queries that are monotonically increasing; the case for queries that are monotonically decreasing is symmetric.

Lemma 6.1. *Let φ be a formula annotated by a monotonic query γ. Then, it holds that $\gamma \equiv_{\mathfrak{M}} \varphi$ iff $\gamma[\top] \Leftrightarrow \gamma[\bot]$ in $\mathfrak{M}$.*

Proof. For the *if* direction, assume that $\gamma[\top] \Leftrightarrow \gamma[\bot]$ in $\mathfrak{M}$. Since γ annotates φ, we know that there exists a subformula ψ of φ such that $\gamma[\psi] = \varphi$. If $\mathfrak{M} \models \gamma[\bot]$, we know by Lemma 2.1 that every formula is a solution to γ in $\mathfrak{M}$. In particular, this implies $\mathfrak{M} \models \gamma[\psi]$, that is, $\mathfrak{M} \models \varphi$. Otherwise, if $\mathfrak{M} \not\models \gamma[\bot]$, then $\mathfrak{M} \not\models \gamma[\top]$ either. Thus, by Lemma 2.1, we know that γ has no solution in $\mathfrak{M}$. In particular, this implies $\mathfrak{M} \not\models \gamma[\psi]$, that is, $\mathfrak{M} \not\models \varphi$. Hence, for every formula θ it holds that $\gamma[\theta] \Leftrightarrow \varphi$ in $\mathfrak{M}$, that is, $\gamma \equiv_{\mathfrak{M}} \varphi$.

For the *only if* direction, assume that $\gamma \equiv_{\mathfrak{M}} \varphi$. Thus, for every formula θ it holds that $\gamma[\theta] \Leftrightarrow \varphi$ in $\mathfrak{M}$. In particular, this implies $\gamma[\top] \Leftrightarrow \varphi$ and $\gamma[\bot] \Leftrightarrow \varphi$ in $\mathfrak{M}$. Hence, we have $\gamma[\top] \Leftrightarrow \gamma[\bot]$ in $\mathfrak{M}$. □

The following theorem can be proved by using Lemma 2.1 and Lemma 6.1. In clear analogy to [KV99], it enables us to reduce vacuity detection with respect to *monotonic* queries to a single model checking call.

Theorem 6.1. *Let φ be a formula annotated by a monotonic query γ. Then, the model $\mathfrak{M}$ satisfies φ strong γ-vacuously iff $\mathfrak{M} \models \gamma[\bot]$.*

Proof. For the *if* direction, assume that $\mathfrak{M} \models \gamma[\bot]$. Since γ annotates φ, we know that there exists a subformula ψ of φ such that $\gamma[\psi] = \varphi$. Moreover, since $\mathfrak{M} \models \gamma[\bot]$, we know by Lemma 2.1 that every formula is a solution to γ in $\mathfrak{M}$. In particular, this implies $\mathfrak{M} \models \gamma[\psi]$, that is, $\mathfrak{M} \models \varphi$. Hence, for every formula θ it holds that $\gamma[\theta] \Leftrightarrow \varphi$ in $\mathfrak{M}$, that is, $\gamma \equiv_{\mathfrak{M}} \varphi$. So we have $\mathfrak{M} \models \varphi$ and $\gamma \equiv_{\mathfrak{M}} \varphi$, i.e., $\mathfrak{M}$ satisfies φ strong γ-vacuously.

For the *only if* direction, assume that $\mathfrak{M} \models \varphi$ and $\gamma \equiv_{\mathfrak{M}} \varphi$. Since γ annotates φ, we know that there exists a subformula ψ of φ such that $\gamma[\psi] = \varphi$. Hence, since $\mathfrak{M} \models \varphi$, we trivially have $\mathfrak{M} \models \gamma[\psi]$. Thus, by Lemma 2.1, we obtain $\mathfrak{M} \models \gamma[\top]$. Moreover, since $\gamma \equiv_{\mathfrak{M}} \varphi$, we know by Lemma 6.1 that $\gamma[\top] \Leftrightarrow \gamma[\bot]$ in $\mathfrak{M}$. So we have $\mathfrak{M} \models \gamma[\top]$ and $\gamma[\top] \Leftrightarrow \gamma[\bot]$ in $\mathfrak{M}$. Hence, it holds that $\mathfrak{M} \models \gamma[\bot]$. □

An immediate consequence of this theorem is the following corollary.

Corollary 6.1. *Let φ be a formula annotated by a monotonic query γ. Then, checking whether a model $\mathfrak{M}$ satisfies φ strong γ-vacuously can be done in time $\mathcal{O}(C_{\mathfrak{M}}(|\gamma[\bot]|))$, where $C_{\mathfrak{M}}(k)$ denotes the complexity of checking whether $\mathfrak{M}$ satisfies a formula of length k.*

The key observation in the above theorem is that checking formula $\gamma[\bot]$ in $\mathfrak{M}$ can be seen as the computation of the strongest solutions to query γ in $\mathfrak{M}$. If there is a single strongest solution equivalent to $\bot$, then $\mathfrak{M}$ satisfies φ strong γ-vacuously. Hence, we have reduced vacuity detection to query solving. But what can we say if the strongest solutions to γ in $\mathfrak{M}$ are not equivalent to $\bot$? We will answer this question by a generalization of strong vacuity in the following section.

6.3.2 Weak Vacuity

Recall that a subformula ψ does not affect the truth value of φ in $\mathfrak{M}$ iff ψ can be replaced by *any* other formula without changing the truth value of the resulting formula in $\mathfrak{M}$. However, as mentioned by Beer et al. in [BBDER01], this definition of vacuity is sometimes "missing the point". We demonstrate this by an example proposed by Amir Pnueli:

Example 6.4 *(Pnueli [Pnu97]).* Consider the formula $\mathbf{AG}\,\mathbf{AF}\,p$ and let $\mathfrak{M}$ be a model such that $\mathfrak{M} \models \mathbf{AG}\,p$. Then, it trivially holds that $\mathfrak{M} \models \mathbf{AG}\,\mathbf{AF}\,p$. Since it cannot be the case that $\mathfrak{M} \models \mathbf{AG}\,\bot$, we know that $\mathbf{AF}\,p$ affects the truth value of $\mathbf{AG}\,\mathbf{AF}\,p$ in $\mathfrak{M}$, i.e., $\mathfrak{M}$ does not satisfy $\mathbf{AG}\,\mathbf{AF}\,p$ strong γ-vacuously, where $\gamma = \mathbf{AG}\,?$. However, our intuition tells us that $\mathfrak{M}$ satisfies $\mathbf{AG}\,\mathbf{AF}\,p$ vacuously since it holds due to a trivial reason, namely because the stronger formula $\mathbf{AG}\,p$ holds in $\mathfrak{M}$. In terms of temporal logic queries, this means that $\mathfrak{M} \models \gamma[\mathbf{AF}\,p]$ holds vacuously because (i) $\mathfrak{M} \models \gamma[p]$, (ii) γ is monotonic, and (iii) $p \Rightarrow \mathbf{AF}\,p$.

The approach proposed by Beer et al. [BBDER01] for solving this problem is to refine the standard definition of vacuity in such a way that formulas as in the example above are identified as vacuously satisfied: "Instead of checking whether a subformula can be replaced by *any* other subformula, we will check whether it can be replaced by *some* 'simpler' formula." They did not give a formal definition of the term "simpler" but they presented some examples: p is simpler than $\mathbf{AF}\,p$, $\mathbf{AG}\,p$ is simpler than $\mathbf{AF}\,p$, and $(\mathbf{AG}\,p) \wedge (\mathbf{AF}\,q)$ is simpler than $\mathbf{A}(p\,\mathbf{U}\,q)$. From these examples it is easy to see, especially with the knowledge of temporal logic queries in mind, that "simpler" means stronger with respect to logical implication, i.e., $p \Rightarrow \mathbf{AF}\,p$, $\mathbf{AG}\,p \Rightarrow \mathbf{AF}\,p$, and $(\mathbf{AG}\,p) \wedge (\mathbf{AF}\,q) \Rightarrow \mathbf{A}(p\,\mathbf{U}\,q)$.

Our intuition to refine vacuity is therefore to define a formula to be vacuously satisfied if there is a subformula that can be replaced by a stronger formula without affecting the truth value. Finding such stronger formulas can be done by solving temporal logic queries. In fact, we will show that it suffices to compute the minimal solutions to a query. However, as already mentioned in Section 4.2, such minimal solutions do not need to exist in general. Moreover, when computing some stronger formulas (not necessarily minimal) they may not be interesting in the sense that they do not justify the truth value of the original formula by a *trivial* reason. Therefore, we have to restrict the set of potential solutions to a set of user-selected formulas that are considered to be interesting for detecting vacuity – we call the elements of this set *vacuity causes.*

Definition 6.7 (Vacuity causes). A set of *vacuity causes* Θ is a poset with respect to logical implication of formulas in a given logic such that every subset of Θ has a finite number of minimal and maximal elements.

In the following, we will implicitly assume that $\{\bot, \top\} \subseteq \Theta$ in order to guarantee that strong vacuity is a special case of our generalization to weak vacuity. Important natural examples of vacuity causes are:

- **Classical vacuity causes** $\{\bot, \top\}$, which yields strong vacuity.
- **Propositional vacuity causes** (e.g., $p \Rightarrow \mathbf{AF}\, p$)
- **Bounded vacuity causes**, i.e., formulas with a maximal nesting depth of temporal operators. In particular:
  - **Local vacuity causes**, where the next operator is the only allowed temporal operator (e.g., $\mathbf{AX}(p \vee \mathbf{AX}\, p) \Rightarrow \mathbf{AF}\, p$).
  - **Invariants** (e.g., $\mathbf{AG}\, p \Rightarrow p$)

The following definition formalizes our generalization by weakening the requirement $\mathfrak{M} \models \gamma[\bot]$ in Theorem 6.1 accordingly. Note that the definition is *parameterized* by a set of vacuity causes.

Definition 6.8 (Weak vacuity). Let φ be a formula annotated by a monotonic query γ such that $\varphi = \gamma[\psi]$, and let Θ be a set of vacuity causes. Then, the model $\mathfrak{M}$ satisfies φ *weak* γ*-vacuously* with vacuity causes in Θ iff

1. $\mathfrak{M} \models \varphi$ and $\psi \Leftrightarrow \bot$ or
2. there exists $\theta \in \Theta$ such that $\mathfrak{M} \models \gamma[\theta]$ and $\theta \Rightarrow \psi$, but $\psi \not\Rightarrow \theta$.

Remark 6.1. Note that the definition of weak vacuity makes only sense for monotonic queries. Moreover, note that if (1) holds true, no solution strictly weaker than ψ can exist, since $\theta \Rightarrow \bot$ is true only for $\theta \Leftrightarrow \bot$.

A naive algorithm for detecting weak vacuity according to Definition 6.8 can be formulated as in Algorithm 3. However, in order to detect weak vacuity, it suffices to compute the strongest resp. minimal solutions to γ in $\mathfrak{M}$ as stated in the following theorem. The restriction to minimal solutions does not only reduce the computation effort but also provides more compact information on the causes of weak vacuity.

Algorithm 3 Detecting weak vacuity

```
1  Select a subformula ψ of φ.
2  Define γ such that φ = γ[ψ].
3  if 𝔐 ⊨ φ and ψ ⇔ ⊥ then          // Special case: ψ ⇔ ⊥
4     output ⊥ ;                    // Vacuity cause: ⊥
5  elsif ψ ⇏ ⊥ then                 // Regular case: ψ ⇏ ⊥
6     for all θ ∈ sol(𝔐, γ) ∩ Θ do
7        if θ ⇒ ψ and ψ ⇏ θ then
8           output θ ;               // Vacuity cause: θ
```

Theorem 6.2. *Let φ be a formula annotated by a monotonic query γ and Θ be a set of vacuity causes. Further, let $\mathcal{M}$ be the set of minimal solutions to γ in a model $\mathfrak{M}$ restricted to elements in Θ. Then, $\mathfrak{M}$ satisfies φ weak γ-vacuously with vacuity causes in Θ iff $\mathfrak{M}$ satisfies φ weak γ-vacuously with vacuity causes in $\mathcal{M}$.*

Proof. Since condition (1) of weak vacuity (cf. Definition 6.8) is trivially equivalent in both cases, it remains to show the equivalence of condition (2).

For the *if* direction, assume that there exists $\mu \in \mathcal{M}$ such that $\mathfrak{M} \models \gamma[\mu]$ and $\mu \Rightarrow \psi$, but $\psi \not\Rightarrow \mu$. Hence, since $\mathcal{M} \subseteq \Theta$, we trivially obtain that $\mathfrak{M}$ satisfies φ weak γ-vacuously with vacuity causes in Θ.

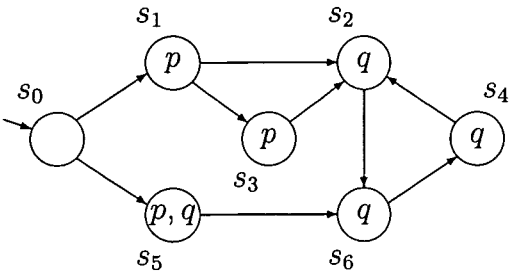


Figure 6.1: Weak vacuity example

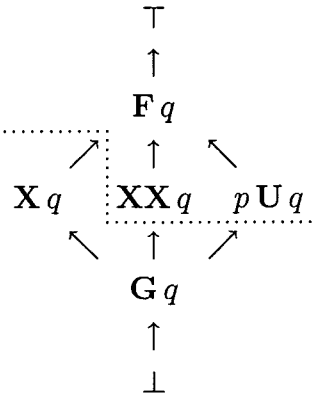


Figure 6.2: Vacuity causes

For the *only if* direction, assume that there exists a formula $\theta \in \Theta$ such that $\mathfrak{M} \models \gamma[\theta]$ and $\theta \Rightarrow \psi$, but $\psi \not\Rightarrow \theta$. Since $\mathcal{M} \subseteq \Theta$ is the set of minimal solutions to γ in $\mathfrak{M}$ and θ is a solution to γ in $\mathfrak{M}$, we know that there exists $\mu \in \mathcal{M}$ such that $\mu \Rightarrow \theta$. Thus, since $\theta \Rightarrow \psi$, we obtain $\mu \Rightarrow \psi$. On the other hand, if $\psi \Rightarrow \mu$, then $\psi \Rightarrow \theta$ since $\mu \Rightarrow \theta$. This, however, contradicts $\psi \not\Rightarrow \theta$. Hence, there exists $\mu \in \mathcal{M}$ such that $\mathfrak{M} \models \gamma[\mu]$ and $\mu \Rightarrow \psi$ but $\psi \not\Rightarrow \mu$, i.e., $\mathfrak{M}$ satisfies φ weak γ-vacuously with vacuity causes in $\mathcal{M}$. □

The following corollary states an immediate practical application of the previous theorem in Algorithm 3.

Corollary 6.2. *Line 6 in Algorithm 3 can be replaced by*

$$\textbf{for all } \theta \in \min_{\Rightarrow}(\mathrm{sol}(\mathfrak{M}, \gamma) \cap \Theta) \textbf{ do}.$$

Remark 6.2. It is easy to see that the computation of solutions according to Line 6 in Algorithm 3 can be optimized in such a way that $\mathrm{sol}(\mathfrak{M}, \gamma)$ directly computes the set of minimal solutions within Θ. Moreover, note that only those solutions computed in Line 6 are relevant that meet the requirements of Line 7. Hence, formula ψ can also be used to reduce the search space during computation in Line 6.

Example 6.5. Consider the Kripke structure $\mathfrak{K}$ shown in Figure 6.1 and the poset Θ of vacuity causes shown in Figure 6.2. Further, let $\varphi = \mathbf{XF}\,q$

be the LTL formula for which we want to check vacuity with respect to subformula $\psi = \mathbf{F}\, q$ in $\mathfrak{K}$. Thus, we define $\gamma = \mathbf{X}\,?$. It is easy to see that $\mathfrak{K} \models \varphi$ and that φ does not hold strong γ-vacuously in $\mathfrak{K}$ since $\mathfrak{K} \not\models \gamma[\bot]$ (cf. Theorem 6.1).

Now, in order to check weak vacuity, let us consider the solutions to γ in $\mathfrak{K}$. It is easy to see that $\mathrm{sol}(\mathfrak{K}, \gamma) \cap \Theta = \{\mathbf{XX}\, q, p\, \mathbf{U}\, q, \mathbf{F}\, q, \top\}$, which are the elements above the dotted line in Figure 6.2. This is also the set computed in Line 6 in Algorithm 3. According to Theorem 6.2, however, it suffices to consider the set $\min_{\Rightarrow}(\mathrm{sol}(\mathfrak{K}, \gamma) \cap \Theta) = \{\mathbf{XX}\, q, p\, \mathbf{U}\, q\}$. Thus, it remains to check the weak vacuity conditions of Line 7 in Algorithm 3, which are satisfied by both formulas. Hence, the two vacuity causes $\mathbf{XX}\, q$ and $p\, \mathbf{U}\, q$ indeed cause weak γ-vacuity of $\mathbf{XF}\, q$ in $\mathfrak{K}$.

Finally, it remains to show how to extract non-vacuity witnesses from the Kripke structure under consideration, i.e., how to obtain a substructure that proves that the given formula is *not* vacuously satisfied in the structure.

6.3.3 Witness Construction

If a formula is not satisfied by the model, the model checker returns a counterexample, which helps the user to correct the error in the model or the specification. On the other hand, if the formula is satisfied by the model, standard model checkers do not return a witness, which would also be helpful for the user to verify that the specification holds in the intended way. In particular, after the vacuity detection process, where all errors causing vacuous satisfaction should have been corrected, a witness would prove that the specification holds indeed non-vacuously. The generation of such vacuity witnesses is discussed in [BBDER97, KV99, BBDER01, GC04b]. In this section, we reformulate the witness construction of Kupferman and Vardi [KV99] in terms of temporal logic queries in order to obtain strong vacuity witnesses. Afterwards, we show how to extend this approach in order to obtain also weak vacuity witnesses.

At first, let us recall the relevant definitions of Beer et al. [BBDER01]. One of the main difficulties in witness construction is to construct an *interesting* witness, i.e., a witness that is as small as possible. To this aim, a pre-order on models resp. witnesses is necessary.

Definition 6.9 (Pre-order on models). Given a logic $\mathcal{L}$, we define the natural *pre-order* $\prec_{\mathcal{L}}$ of $\mathcal{L}$ on the set of models: $\mathfrak{M}' \prec_{\mathcal{L}} \mathfrak{M}$ iff for all $\varphi \in \mathcal{L}$ it holds that $\mathfrak{M} \models \varphi \Rightarrow \mathfrak{M}' \models \varphi$.

Natural pre-orders on models of the temporal logics LTL, CTL, CTL*, ACTL, and ACTL* can be found in [BBDER01]. Since it is clear now that such a pre-order on models depends on the logic under consideration, we will omit the subscript $\mathcal{L}$ and simply write $\prec$ in the following.

Now, we are able to define what we understand by an interesting witness. Note that we consider the more general concept of witnesses with respect to a set of queries and not with respect to a single query. This is because a witness that proves non-vacuity with respect to a set of queries simultaneously contains much more information than the individual witnesses with respect to each single query in this set. The following definition is a straight forward adaption of Beer et al. [BBDER01].

Definition 6.10 (Interesting witness[1]). Let φ be a formula and Γ be a set of queries that annotate φ. Further, let $\mathfrak{M}$ be a model and Θ be a set of vacuity causes. Then, the model $\mathfrak{C} \prec \mathfrak{M}$ is an *interesting strong (weak) Γ-vacuity witness* of φ in $\mathfrak{M}$ (with vacuity causes in Θ) iff $\mathfrak{C}$ is a minimal model with respect to $\prec$ such that for all $\gamma \in \Gamma$ it holds that $\mathfrak{C}$ does not satisfy φ strong (weak) γ-vacuously (with vacuity causes in Θ).

Note that there exists an interesting γ-vacuity witness of a formula φ in a model $\mathfrak{M}$ iff φ does not hold γ-vacuously in $\mathfrak{M}$ [BBDER01]. However, note further that a single interesting witness with respect to a set of queries does not need to exist as it is shown in the following example.

Example 6.6 *(Beer et al. [BBDER01]).* There cannot exist a single interesting strong Γ-vacuity witness of $\varphi = p \vee q$ in any model, where $\Gamma = \{\gamma_1 = p \vee ?, \gamma_2 = ? \vee q\}$. For the sake of contradiction, assume that there exists such a witness $\mathfrak{C}$. Then, we know that $\mathfrak{C} \models \varphi$, but $\mathfrak{C} \not\models \gamma_1[\bot]$ and $\mathfrak{C} \not\models \gamma_2[\bot]$. However, since $\mathfrak{C} \not\models \gamma_1[\bot]$ implies $\mathfrak{C} \not\models p$ and $\mathfrak{C} \not\models \gamma_2[\bot]$ implies $\mathfrak{C} \not\models q$, we have $\mathfrak{C} \not\models p \vee q$, which contradicts $\mathfrak{C} \models \varphi$.

[1] A restricted definition of an interesting witness is presented in [KV99] where only paths are considered to be simple enough to be interesting. However, there may exist simple witnesses (cf. Clarke et al. [CJLV02]) in cases where no path witnesses exist.

Hence, if a single witnesses with respect to a set of queries does not exist, we have to split the set and construct a witness for each subset. Thus, in the worst case, we have to construct a witness for each query.

Kupferman and Vardi [KV99] defined a witness formula in order to generate an interesting witness. The following definition adapts their construction in terms of temporal logic queries.

Definition 6.11 (Strong vacuity witness formula). Let φ be a formula and Γ be a set of monotonic queries that annotate φ. Then, the *strong* Γ-*vacuity witness formula* of φ is given by

$$witness_{strong}(\varphi, \Gamma) \quad = \quad \varphi \wedge \bigwedge_{\gamma \in \Gamma} \neg\gamma[\bot].$$

It is easy to see by the proof of Theorem 7 in [KV99], that a minimal (wrt. the pre-order on models) counterexample to $\neg witness_{strong}(\varphi, \Gamma)$ in a model $\mathfrak{M}$ is an interesting strong Γ-vacuity witness of φ in $\mathfrak{M}$. Now, let us define a generalization of strong vacuity witness formulas in order to generate an interesting vacuity witness concerning weak vacuity.

Definition 6.12 (Weak vacuity witness formula). Let φ be a formula and Γ be a set of monotonic queries that annotate φ. Further, let Θ be a set of vacuity causes. Then, the *weak* Γ-*vacuity witness formula* of φ with vacuity causes in Θ is given by

$$witness_{weak}(\varphi, \Gamma, \Theta) \quad = \quad \varphi \wedge \bigwedge_{\substack{\gamma \in \Gamma, \varphi = \gamma[\psi] \\ \nu \in \max_{\Rightarrow}(\{\theta \in \Theta \mid \theta \Rightarrow \psi, \psi \not\Rightarrow \theta\})}} \neg\gamma[\nu].$$

The witness formula consists of a conjunction of the formula φ itself, because φ must be true on the witness, and of formulas that must not be true on the witness, namely formulas where the relevant subformulas ψ have been replaced by some stronger vacuity causes. The following theorem is an analogous result to Theorem 7 in [KV99].

Theorem 6.3. *Let φ be a formula and Γ be a set of monotonic queries that annotate φ. Further, let Θ be a set of vacuity causes. Then, a minimal (wrt. the pre-order on models) counterexample to $\neg witness_{weak}(\varphi, \Gamma, \Theta)$ in a model $\mathfrak{M}$ is an interesting weak Γ-vacuity witness of φ in $\mathfrak{M}$ with vacuity causes in Θ.*

Proof. Since we reduce witness construction to counterexample construction by using the counterexample returned by a model checking tool, we assume that the minimality of the witness (cf. Definition 6.10) is guaranteed by the underlying counterexample techniques. Hence, it remains to show that for all $\gamma \in \Gamma$ it holds that φ does not hold weak γ-vacuously in $\mathfrak{C}$. To this aim, let $\mathfrak{C}$ be a counterexample of $\neg witness_{weak}(\varphi, \Gamma, \Theta)$ in $\mathfrak{M}$. Then, $\mathfrak{C}$ obviously satisfies $witness_{weak}(\varphi, \Gamma, \Theta)$ and therefore $\mathfrak{C} \models \varphi$. Now, assume that there exists a query $\gamma \in \Gamma$ such that $\mathfrak{C}$ satisfies φ weak γ-vacuously with vacuity causes in Θ. Since γ annotates φ, we know that there exists a subformula ψ of φ such that $\gamma[\psi] = \varphi$. Consequently, there exists $\theta \in \Theta$ such that $\mathfrak{C} \models \gamma[\theta]$ and $\theta \Rightarrow \psi$, but $\psi \not\Rightarrow \theta$. Hence, since Θ is a set of vacuity causes, there must exist a maximal $\nu \in \Theta$ such that $\theta \Rightarrow \nu$, $\nu \Rightarrow \psi$, and $\psi \not\Rightarrow \nu$. Thus, by the monotonicity of γ, we know that $\mathfrak{C} \models \gamma[\nu]$ and $\nu \Rightarrow \psi$, but $\psi \not\Rightarrow \nu$, which contradicts $\mathfrak{C} \models witness_{weak}(\varphi, \Gamma, \Theta)$. It follows that for all $\gamma \in \Gamma$ it holds that $\mathfrak{C}$ does not satisfy φ weak γ-vacuously with vacuity causes in Θ, i.e., $\mathfrak{C}$ is an interesting weak Γ-vacuity witness of φ in $\mathfrak{M}$ with vacuity causes in Θ. □

Example 6.7. Consider the Kripke structure $\mathfrak{K}$ shown in Figure 6.1 and the poset Θ of vacuity causes shown in Figure 6.2. In order to compare strong and weak vacuity witnesses, we need a formula that holds neither strong nor weak vacuously in $\mathfrak{K}$ with vacuity causes in Θ. From Example 6.5, we know that, e.g., $\mathbf{XXX}\,q$ does not hold weak γ-vacuously (where $\gamma = \mathbf{X}\,?$) and therefore not strong γ-vacuously. Thus, let $\varphi = \mathbf{XXX}\,q$ be the LTL formula for which we want to construct vacuity witnesses. Since we consider LTL formulas, counterexamples and vacuity witnesses are paths (cf. Beer et al. [BBDER01]). There exist three paths in $\mathfrak{K}$: $\pi_1 = s_0, s_1, s_2, s_6, s_4, \ldots$, $\pi_2 = s_0, s_1, s_3, s_2, s_6, s_4, \ldots$, and $\pi_3 = s_0, s_5, s_6, s_4, s_2, \ldots$.

It is easy to see that all three paths are interesting strong γ-vacuity witnesses of φ in $\mathfrak{M}$, i.e., all three paths are minimal and $\pi_i \not\models \gamma[\bot]$ for all $1 \leq i \leq 3$. However, π_3 cannot be a weak γ-vacuity witness because $\pi_3 \models \gamma[\mathbf{G}\,q]$, i.e., there exists a formula stronger than $\mathbf{XX}\,q$ in Θ that is a solution to γ on π_3. More formally, π_3 is not a counterexample to $\neg witness_{weak}(\varphi, \gamma, \Theta)$ in $\mathfrak{K}$ (cf. Theorem 6.3), where $witness_{weak}(\varphi, \gamma, \Theta) = \varphi \wedge \neg\gamma[\mathbf{G}\,q]$. On the other hand, it is easy to see that π_1 and π_2 are interesting weak γ-vacuity witnesses of φ in $\mathfrak{K}$ with vacuity causes in Θ.

The remaining question for detecting weak vacuity is how to compute the set of minimal solutions or at least one minimal solution that satisfies the conditions of Theorem 6.2. To this aim, the query solving algorithms presented in Chapter 5 can be used. Finally, let us remark that there are two aspects of query solving for vacuity detection that potentially reduce its complexity: First, the set of formulas that are taken into account as solutions can be restricted to the set of vacuity causes. Second, it suffices to find a (minimal) solution that is *stronger than a given formula.* These constraints restrict the search space and therefore we believe that computing solutions for vacuity detection can be done more efficiently.

6.4 Related Work

The following summarizes the most important publications in vacuity detection. Those of them that are directly related to our work have already been cited at the corresponding positions in the text. The other ones are only described for the sake of completeness.

Beer et al. [BBDER97] is to our best knowledge the first paper in which automatic vacuity detection was investigated. They restricted their considerations to the syntactic class w-ACTL of witnessable ACTL formulas and proved that for every w-ACTL formula φ there is a formula $w(\varphi)$ such that both φ and $w(\varphi)$ are true in a model $\mathfrak{M}$ iff φ holds vacuously in $\mathfrak{M}$. In addition, they showed that if $w(\varphi)$ does not hold in $\mathfrak{M}$, then any counterexample is a non-vacuity witness to φ in $\mathfrak{M}$.

Kupferman and Vardi [KV99] noticed that every occurrence of a subformula of a CTL* formula φ occurs either positive (i.e., under an even number of negations) or negative (i.e., under an odd number of negations) in φ. Together with their restriction to detect vacuity with respect to *occurrences* of subformulas, this enabled them to reduce vacuity detection to model checking of formulas in which some subformula is replaced by a constant truth value. They also showed how to generate non-vacuity witnesses of linear witnessable formulas φ by defining a witness formula $witness(\varphi)$ such that a counterexample to $\neg witness(\varphi)$ in a model $\mathfrak{M}$ is an interesting witness to φ in $\mathfrak{M}$.

Beer et al. [BBDER01] generalized the approach of Kupferman and Vardi [KV99] to vacuity detection in any logic with polarity (i.e., every subformula occurs either positive or negative) and to witness construction in any logic with a pre-order on the models.

Armoni et al. [AFF+03] investigated several kinds of vacuity semantics (formula semantics, structure semantics, trace semantics) for LTL by using universal propositional quantification. The focus of their work lay on vacuity detection with respect to subformulas with multiple occurrences (some of them under an even and some of them under an odd number of negations). It was shown that all these semantics are equivalent for vacuity with respect to subformulas of pure (i.e., either positive or negative) polarity.

Gurfinkel and Chechik [GC04a, GC04b] dealt with four-valued vacuity (vacuously true, non-vacuously true, non-vacuously false, vacuously false) and with mutual influence of subformulas when checking vacuity with respect several subformulas simultaneously ("mutual vacuity"). In this way, they showed how to encode different degrees of vacuity into a multi-valued logic. Moreover, in [GC04a], they studied generalizations of the vacuity semantics of Armoni et al. [AFF+03] by moving from LTL to CTL*, i.e., from traces to trees.

Bustan et al. [BFG+] recently investigated vacuity detection in the logic RELTL, an extension of LTL by a regular layer. They defined a formula to be regularly vacuous if there exists a regular subexpression e such that the resulting formula after replacing e by a universal quantified second-order interval variable remains true. Analogous to Kupferman and Vardi [KV99], if the subexpression has pure polarity, then regular vacuity detection can be reduced to regular model checking.

6.5 Summary

If a specification is not satisfied by the model, common model checkers return a counterexample. Otherwise, if a specification is satisfied by the model, it is in general unknown whether the specification holds *vacuously*, i.e., due to a trivial reason. Practical experience has shown that vacuously

satisfied specifications often hint at a real problem either in the model or in the specification. Therefore, detecting vacuity is an important and challenging task in verification and validation.

Kupferman and Vardi [KV99] showed that in certain cases vacuity detection can be reduced to model checking. In particular, a formula is vacuously satisfied if the truth value of the formula remains unchanged in a given model when replacing any (purely occurring) subformula by a constant truth value. For example, the formula $\varphi = \mathbf{AX}(p \vee \mathbf{AF}\, q)$ is vacuously satisfied with respect to $\mathbf{AF}\, q$ if the truth value of $\mathbf{AX}(p \vee \bot) = \mathbf{AX}\, p$ remains unchanged. This procedure can be easily reduced to the computation of minimal solutions to temporal logic queries. For example, if $\bot$ is a minimal solution to the query $\gamma = \mathbf{AX}(p \vee ?)$, then φ is vacuously satisfied with respect to $\mathbf{AF}\, q$.

The reduction of vacuity detection to query solving enables us to extend the classical notion of vacuity. For example, if q but not $\bot$ is a minimal solution to γ, then φ is not vacuously satisfied with respect to $\mathbf{AF}\, q$ in the classical sense although it holds due to a trivial reason, namely because the stronger formula $\gamma[q]$ holds in the model. Hence, the classical notion of vacuity can be seen as an extreme case of vacuity, where the subformula collapses to a constant truth value. This can be generalized by a parameterization through a user-defined class of *vacuity causes*. Then, a specification is vacuously satisfied if a subformula collapses to a vacuity cause.

Chapter 7

Conclusion and Outlook

7.1 Summary

In this thesis, we have investigated several approaches in order to reason about specifications in general and temporal logic specifications in particular. Since specifications are meant to describe the system behavior as precisely as possible, reasoning about specifications provides additional information to support verification and validation engineers in their task to decide whether the system under consideration satisfies the specified and intended requirements. In particular, we have studied three main questions:

1. Given the specifications satisfied by two systems, what can we say about the system obtained by composing these systems?

2. Given a system model and an incomplete temporal logic specification, how can the given specification be efficiently completed such that it is satisfied by the model?

3. Given a system model that satisfies its specification, does the model satisfy the specification in the intended way?

In Chapter 3, we dealt with the first question. In particular, we presented the circular counterpart of the classical cut rule in logic calculi. We proved that our *circular cut rule* is sound and we demonstrated how it can be used in mutual inductive proofs. The investigation of such an inference rule was motivated by our proofs in Chapter 4, where circular dependent lemmas on properties of temporal logic query languages were systematically composed in order to obtain stronger results by reducing the number of assumptions. In this way, we obtained some of our main results. Moreover, in analogy

to the circular cut rule, we presented a circular composition rule for composing arbitrary circular dependent systems. Compositional reasoning had already been investigated in the literature. Our rule formalizes the common idea of all these approaches on an abstract level. By a simple example, we demonstrated how the circular composition rule can be successively applied in order to obtain specifications satisfied by composed systems from specifications satisfied by their components.

In Chapter 4 and Chapter 5, we dealt with the second question. In particular, in Chapter 4, we investigated several properties of temporal logic queries and their relationship. Afterwards, we presented our syntactic characterization of exact LTL queries in form of a deterministic, context-free template grammar capturing all monotonic single-variable LTL queries. The query templates derived in this grammar were divided into two classes: LTLQ^x and $\overline{\mathrm{LTLQ}^x}$. At first, we proved that all LTL queries obtained by instantiating the templates in LTLQ^x are collecting and therefore exact. This was done by composing circular dependent auxiliary results that were based on the properties *strong collecting*, *boundary collecting*, *intermediate collecting*, and *weak collecting*. Then, we proved that for all templates in $\overline{\mathrm{LTLQ}^x}$ there exists a simple instantiation that is not collecting and therefore not exact. This was done by constructing a counterexample path to the collecting property for all such instantiations.

Afterwards, we presented our large fragment of exact CTL queries in form of a deterministic, context-free template grammar capturing all monotonic single-variable CTL queries. In analogy to the case of LTL, the query templates derived in this grammar were divided into two classes: CTLQ^x and $\overline{\mathrm{CTLQ}^x}$. We proved that all CTL queries obtained by instantiating the templates in CTLQ^x are collecting and therefore exact. As in the case of LTL, this was done by composing circular dependent auxiliary results that were based on the properties *strong collecting*, *boundary collecting*, *intermediate collecting*, and *weak collecting*. Unfortunately, we were not able to prove the maximality of CTLQ^x as we have done for LTLQ^x. In fact, we showed that there exist simple queries in $\overline{\mathrm{CTLQ}^x}$ that are exact, that is, CTLQ^x is not maximal. We argued that a syntactic characterization in the case of CTL is much more difficult.

Further, in Chapter 5, we presented and analyzed several algorithms for solving temporal logic queries. At first, we dealt with Chan's symbolic query solving algorithm. To this aim, we divided the queries in our

fragment CTLQ^x into universal and existential occurring ones. Then, we showed that all existential occurring queries are *intermediate collecting*, which enabled us to reduce non-determinism in the sense that existential choices can be eliminated when solving such queries. In this way, we were able to prove the correctness of Chan's algorithm. Moreover, based on our insights, we were able to extend Chan's algorithm such that it is applicable to the whole fragment CTLQ^x and not only to its valid subset. Finally, we showed how several algorithms for solving temporal logic queries can be modified in order to compute also non-propositional solutions.

In Chapter 6, we dealt with the third question. In particular, we redefined the classical notion of vacuity by temporal logic queries in such a way that a subformula of the formula to be checked is substituted by the placeholder. Classical vacuity (*strong vacuity*) detection corresponds then to deciding whether the constant truth value *false* (resp. *true*) is a solution to such a query. We interpreted this procedure as finding a solution that implies the original subformula, which led naturally to the generalized notion of *weak vacuity*. In this way, we were able to solve a problem posed by Amir Pnueli. He presented a specification that is trivially satisfied but does not fall under the classical notion of vacuity; however, it falls under our notion of weak vacuity. Finally, we described how to obtain weak vacuity witnesses.

7.2 Open Questions

In the following, we conclude this thesis by listing some remaining open questions that point out the way for further research:

1. Recall that we proved the soundness of both the circular cut rule and the circular composition rule. However, we did not consider completeness which is also a natural and important property of inference rules. Moreover, it would be interesting to know which proof-theoretic implications our circular cut rule involves.

2. Both the circular cut rule and the circular composition rule can be used to compose proofs and systems respectively that depend on each other in a circular manner. The kind of circularity, however, is fixed in our inference rules. It would therefore be interesting to know if there exist other kinds of circular dependencies that can be formalized

in this way and that are not captured by our composition rule. In addition, the exact relationship of our results to other compositional reasoning approaches should be clarified.

3. Since our basic results on temporal logic queries were proved purely semantically, they hold also for other logic based query languages. Thus, a natural question concerns applications and further developments of our results in other logic formalisms. For example, what can we say about exact first-order and exact monadic second-order logic queries? In addition, what can we say about the expressive power of exact queries in such formalisms?

4. Recall that our syntactic characterization of exact LTL queries is based on templates. It is therefore possible to find refined characterizations that are based on refined templates. Moreover, it is possible that there exists a simple grammar that characterizes all exact LTL queries modulo logical equivalence.

5. The task of finding a syntactic characterization of exact CTL queries is certainly the most obvious starting point for future research. Such a characterization has also immediate practical relevance, since our extended Chan algorithm exploits the properties identified in this way. Further possibilities to investigate exact CTL queries follow in analogy to the case of LTL as described in item 4.

6. For both LTL and CTL, it would be interesting to investigate queries with several placeholders. Of course, our basic results hold for an arbitrary number of placeholders; our syntactic fragments, however, are restricted to a single placeholder. Also most of the presented algorithms require a single occurrence of the placeholder.

7. We described how the Chan resp. extended Chan algorithm works and which properties it exploits in order to efficiently compute an exact solution. It is possible that also other algorithms can be optimized by exploiting similar properties. In addition, the principle on which Chan's algorithm is based on (i.e., eliminating existential choices) may also be useful in other fields.

8. A more practical task is to implement the presented query solving algorithms and to compare their efficiency by several benchmark tests. To our best knowledge there exist no implementations except of the algorithm by Chechik et al. described in Section 5.3.2. We believe that the Chan resp. extended Chan algorithm is more efficient than the other algorithms for queries on which it is applicable. Moreover, an implementation of our weak vacuity detection algorithm in form of an experimental tool would be useful.

Appendix A

Omitted Proofs for LTLQ^x

A.1 Proof of Exactness

Proof of Lemma 4.6

Lemma 4.6. *Let $\gamma \in LTLQ^2$. Suppose that every subquery in $LTLQ^1$ is weak collecting. Then, γ is intermediate collecting.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\theta$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we trivially obtain the assertion by $\pi^k \models \gamma[\bot]$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\theta)$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$. Thus, it holds that $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we trivially obtain the assertion by $\pi^k \models \gamma[\bot]$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{U}\,\theta$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \vee \bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. If $\pi \models \theta$ or $\pi^n \models \theta$, we trivially obtain $\pi \models \gamma[\bot]$ resp. $\pi^n \models \gamma[\varphi \wedge \psi]$. Otherwise, if $\pi \not\models \theta$ and $\pi^n \not\models \theta$, we know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^n \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $r < n$ and $\pi^r \models \bar{\gamma}[\bot]$, which trivially imply $\pi^n \models \gamma[\varphi \wedge \psi]$ and $\pi^r \models \gamma[\bot]$ respectively.

▷ Let $\gamma = \mathbf{X}\bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that $\pi^1 \models \bar{\gamma}[\varphi]$ and $\pi^{n+1} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{n+1} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $r < n$ and $\pi^{r+1} \models \bar{\gamma}[\bot]$, which imply $\pi^n \models \gamma[\varphi \wedge \psi]$ and $\pi^r \models \gamma[\bot]$ respectively.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\theta$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we trivially obtain the assertion by $\pi^k \models \gamma[\bot]$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$ and $\pi^k \models \theta$. Hence, by induction hypothesis, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta\,\mathbf{U}\,\bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, n+l) \leq r < \max(k, n+l)$ and $\pi^r \models \bar{\gamma}[\bot]$. If $r < n$, we trivially obtain the assertion by $\pi^r \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that either $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[n,\max(k,n+l))} \models \theta$ and $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, in both cases we have $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\theta)$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$. Thus, it holds that $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we trivially obtain the assertion by $\pi^k \models \gamma[\bot]$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$.

So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{U}\, \theta$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\mathbf{W}\, \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, \bar{\gamma})$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^n \models \mathbf{G}\,\theta$, we trivially obtain $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, n+l) \leq r < \max(k, n+l)$ and $\pi^r \models \bar{\gamma}[\bot]$. If $r < n$, we trivially obtain the assertion by $\pi^r \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that either $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[n,\max(k,n+l))} \models \theta$ and $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, in both cases we have $\pi^n \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

This concludes the proof. □

Proof of Lemma 4.7

Lemma 4.7. *Let $\gamma \in$ LTLQ^1. Suppose that every subquery in LTLQ^2 is intermediate collecting. Then, γ is weak collecting.*

Proof. Structural induction on γ.

Induction start:

▷ If γ is the placeholder, then γ is trivially weak collecting.

▷ Let $\gamma = \theta \wedge \bar{\gamma}$ such that $\bar{\gamma} \in$ LTLQ^2. It is easy to see that $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\theta \wedge (\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$. Hence, by assumption, we obtain $\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma} \,\mathring{\mathbf{U}}\, \theta = \bar{\gamma} \,\mathbf{U}\, (\bar{\gamma} \wedge \theta)$ such that $\bar{\gamma} \in$ LTLQ^2. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and therefore $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\mathring{\mathbf{U}}\, \bar{\gamma} = \theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma})$ such that $\bar{\gamma} \in$ LTLQ^2. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^l \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, l) \leq r < \max(k, l)$ and $\pi^r \models \bar{\gamma}[\bot]$.

Consequently, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that either $\pi^{[0,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[0,\max(k,l)]} \models \theta$ and $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, in both cases we have $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\bar{\mathbf{U}}\, \bar{\gamma} = \theta \,\mathbf{U}\, (\neg\theta \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \text{LTLQ}^2$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma} \,\mathring{\mathbf{W}}\, \theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, (\bar{\gamma} \wedge \theta))$ such that $\bar{\gamma} \in \text{LTLQ}^2$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Now, we have to distinguish between two cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and therefore $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\varphi \wedge \psi] \,\mathring{\mathbf{U}}\, \theta$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\mathring{\bar{\mathbf{W}}}\, \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \text{LTLQ}^2$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Now, we have to distinguish between two cases: (i) If $\pi \models \mathbf{G}\,\theta$, we trivially obtain $\pi \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^l \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k,l) \leq r < \max(k,l)$ and $\pi^r \models \bar{\gamma}[\bot]$. Consequently, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that either $\pi^{[0,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[0,\max(k,l)]} \models \theta$ and $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, in both cases we have $\pi \models \theta \,\mathring{\bar{\mathbf{U}}}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\bar{\mathbf{W}}\, \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, (\neg\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \text{LTLQ}^2$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Now, we have to distinguish between two cases: (i) If $\pi \models \mathbf{G}\,\theta$, we trivially obtain $\pi \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta \,\bar{\mathbf{U}}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. It is easy to see that $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\theta \wedge (\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$. Hence, by induction hypothesis, we obtain $\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \vee \bar{\gamma}$. It is easy to see that $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\theta \vee (\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$. Hence, by induction hypothesis, we obtain $\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. It is easy to see that $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\mathbf{X}\,(\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$. Hence, by induction hypothesis, we obtain $\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\theta = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta)$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and therefore $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta\,\bar{\mathbf{U}}\,\bar{\gamma} = \theta\,\mathbf{U}\,(\neg\theta \wedge \bar{\gamma})$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{W}}\,\theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta))$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Now, we have to distinguish between two cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and therefore $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta\,\bar{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,(\neg\theta \wedge \bar{\gamma}))$. Suppose that $\pi \models \gamma[\varphi] \wedge \gamma[\psi]$. Now, we have to distinguish between two cases: (i) If $\pi \models \mathbf{G}\,\theta$, we trivially obtain $\pi \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta\,\bar{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \gamma[\varphi \wedge \psi]$.

This concludes the proof. □

Proof of Lemma 4.9

Lemma 4.9. *Let $\gamma \in LTLQ^4$. Suppose that for every subquery $\bar{\gamma} \in LTLQ^5$ it holds that $\mathbf{G}\,\bar{\gamma}$ is weak collecting. Then, $\mathbf{F}\,\gamma$ is boundary collecting.*

Proof. Structural induction on γ.

Induction start:

- ▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\theta$ such that $\bar{\gamma} \in \mathrm{LTLQ}^3 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. Since θ trivially implies $\gamma[\varphi \wedge \psi]$, we obtain $\pi^{n+l} \models \gamma[\varphi \wedge \psi]$ and therefore $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\theta)$ such that $\bar{\gamma} \in \mathrm{LTLQ}^5$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{\max(k,n+l)} \models \mathbf{G}\,\bar{\gamma}[\varphi] \wedge \mathbf{G}\,\bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{\max(k,n+l)} \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^{\max(k,n+l)} \models \gamma[\varphi \wedge \psi]$ and therefore $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exists $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we trivially obtain $\pi^k \models \gamma[\bot]$ and therefore $\pi \models \mathbf{F}\,\gamma[\bot]$. Otherwise, if $k \geq n$, we trivially obtain $\pi^k \models \gamma[\varphi \wedge \psi]$ and therefore $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$.

Induction step:

- ▷ Let $\gamma = \theta \vee \bar{\gamma}$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. So we have $\pi \models \mathbf{F}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{F}\,\bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$ or $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially imply $\pi \models \mathbf{F}\,\gamma[\bot]$ and $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$ respectively. (ii) Otherwise, there exists $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we trivially obtain $\pi^k \models \gamma[\bot]$ and therefore $\pi \models \mathbf{F}\,\gamma[\bot]$. Otherwise, if $k \geq n$, we trivially obtain $\pi^k \models \gamma[\varphi \wedge \psi]$ and therefore $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^{k+1} \models \bar{\gamma}[\varphi]$ and $\pi^{n+l+1} \models \bar{\gamma}[\psi]$. So we have $\pi^1 \models \mathbf{F}\,\bar{\gamma}[\varphi]$ and $\pi^{n+1} \models \mathbf{F}\,\bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^1 \models \mathbf{F}\,\bar{\gamma}[\bot]$ or $\pi^{n+1} \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$, which imply $\pi \models \mathbf{F}\,\gamma[\bot]$ and $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\theta$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. Since θ trivially implies $\gamma[\varphi \wedge \psi]$, we obtain $\pi^{n+l} \models \gamma[\varphi \wedge \psi]$ and therefore $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta\,\mathbf{U}\,\bar{\gamma}$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. So we have $\pi \models \mathbf{F}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{F}\,\bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$ or $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$, which imply $\pi \models \mathbf{F}\,\gamma[\bot]$ and $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \theta\,\mathbf{W}\,\bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,\bar{\gamma})$. Suppose that $\pi \models \mathbf{F}\,\gamma[\varphi]$ and $\pi^n \models \mathbf{F}\,\gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If there exists $k \in \mathbb{N}$ such that $\pi^k \models \mathbf{G}\,\theta$, we trivially obtain $\pi^k \models \gamma[\bot]$ and therefore $\pi \models \mathbf{F}\,\gamma[\bot]$. (ii) Otherwise, there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. So we have $\pi \models \mathbf{F}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{F}\,\bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$ or $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$, which imply $\pi \models \mathbf{F}\,\gamma[\bot]$ and $\pi^n \models \mathbf{F}\,\gamma[\varphi \wedge \psi]$ respectively.

This concludes the proof. □

Proof of Lemma 4.10

Lemma 4.10. *Let $\gamma \in LTLQ^5$. Suppose that for every subquery $\bar{\gamma} \in LTLQ^4$ it holds that $\mathbf{F}\,\bar{\gamma}$ is weak collecting. Then, $\mathbf{G}\,\gamma$ is weak collecting.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \theta\,\mathring{\mathbf{U}}\,\bar{\gamma} = \theta\,\mathbf{U}\,(\theta \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. Consider the formula $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$, which is equivalent to $\mathbf{G}\,(\gamma[\varphi] \wedge \gamma[\psi])$. Since $\gamma[\varphi] \wedge \gamma[\psi]$ implies $\mathbf{F}\,\bar{\gamma}[\varphi] \wedge \mathbf{F}\,\bar{\gamma}[\psi]$, we obtain by assumption $\mathbf{GF}\,\bar{\gamma}[\varphi \wedge \psi]$. On the other hand, it is easy to see that $\mathbf{G}\,\gamma[\varphi]$ implies $\mathbf{G}\,\theta$. Thus, $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ implies $\mathbf{G}\,\theta \wedge \mathbf{GF}\,\bar{\gamma}[\varphi \wedge \psi]$, which is equivalent to $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta\,\mathbf{W}\,\bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,\bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1$. Consider the formula $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$, which is equivalent to $\mathbf{G}\,(\gamma[\varphi] \wedge \gamma[\psi])$. If θ does not hold, we know that $\gamma[\varphi] \wedge \gamma[\psi]$ implies $\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by Lemma 4.8, we obtain $\bar{\gamma}[\varphi \wedge \psi]$. Thus, $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ implies $\mathbf{G}\,(\theta \vee \bar{\gamma}[\varphi \wedge \psi])$, which is equivalent to $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \mathbin{\mathring{\mathbf{W}}} \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \mathbin{\mathbf{U}} (\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1 \cup \mathrm{LTLQ}^3 \cup \mathrm{LTLQ}^4 \cup \mathrm{LTLQ}^6$. Consider the formula $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$. It is easy to see that $\mathbf{G}\,\gamma[\varphi]$ implies $\mathbf{G}\,\theta$. Thus, since $\mathbf{G}\,\theta$ is equivalent to $\mathbf{GG}\,\theta$ and $\mathbf{G}\,\theta$ trivially implies $\gamma[\varphi \wedge \psi]$, we have $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. It is easy to see that $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ is equivalent to $\mathbf{G}\,\theta \wedge (\mathbf{G}\,\bar{\gamma}[\varphi] \wedge \mathbf{G}\,\bar{\gamma}[\psi])$. Hence, by induction hypothesis, we obtain $\mathbf{G}\,\theta \wedge \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which is equivalent to $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. It is easy to see that $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ is equivalent to $\mathbf{XG}\,\bar{\gamma}[\varphi] \wedge \mathbf{XG}\,\bar{\gamma}[\psi]$, which in turn is equivalent to $\mathbf{X}\,(\mathbf{G}\,\bar{\gamma}[\varphi] \wedge \mathbf{G}\,\bar{\gamma}[\psi])$. Hence, by induction hypothesis, we obtain $\mathbf{XG}\,\bar{\gamma}[\varphi \wedge \psi]$, which is equivalent to $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma} \mathbin{\mathring{\mathbf{U}}} \theta = \bar{\gamma} \mathbin{\mathbf{U}} (\bar{\gamma} \wedge \theta)$. It is easy to see that $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ implies $\mathbf{G}\,\bar{\gamma}[\varphi] \wedge \mathbf{G}\,\bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$. On the other hand, it is easy to see that $\mathbf{G}\,\gamma[\varphi]$ implies $\mathbf{GF}\,\theta$. Thus, we know that $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ implies $\mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi] \wedge \mathbf{GF}\,\theta$, which is equivalent to $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma} \mathbin{\mathring{\mathbf{W}}} \theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \mathbin{\mathbf{U}} (\bar{\gamma} \wedge \theta))$. It is easy to see that $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$ implies $\mathbf{G}\,\bar{\gamma}[\varphi] \wedge \mathbf{G}\,\bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$ is equivalent to $\mathbf{GG}\,\bar{\gamma}[\varphi \wedge \psi]$ and $\mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$ trivially implies $\gamma[\varphi \wedge \psi]$, we have $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \mathbin{\mathring{\mathbf{W}}} \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \mathbin{\mathbf{U}} (\theta \wedge \bar{\gamma}))$. Consider the formula $\mathbf{G}\,\gamma[\varphi] \wedge \mathbf{G}\,\gamma[\psi]$. It is easy to see that $\mathbf{G}\,\gamma[\varphi]$ implies $\mathbf{G}\,\theta$. Thus, since $\mathbf{G}\,\theta$ is equivalent to $\mathbf{GG}\,\theta$ and $\mathbf{G}\,\theta$ trivially implies $\gamma[\varphi \wedge \psi]$, we have $\mathbf{G}\,\gamma[\varphi \wedge \psi]$.

This concludes the proof. □

Proof of Lemma 4.12

Lemma 4.12. *Every query in LTLQ^7 is boundary collecting.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \mathbf{F}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^2$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by Corollary 4.2, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, n+l) \leq r < \max(k, n+l)$ and $\pi^r \models \bar{\gamma}[\bot]$. So, if $\pi^r \models \bar{\gamma}[\bot]$, we trivially have $\pi \models \gamma[\bot]$, and if $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, we trivially have $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{F}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$, that is, $\pi \models \mathbf{F}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{F}\,\bar{\gamma}[\psi]$. Hence, by Lemma 4.11, we obtain $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$ or $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \mathbf{G}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1 \cup \mathrm{LTLQ}^2 \cup \mathrm{LTLQ}^5$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Thus, we know that $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi] \wedge \mathbf{G}\,\bar{\gamma}[\psi]$. Hence, by Lemma 4.8 resp. Corollary 4.2 resp. Corollary 4.3, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Thus, we know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^n \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi \models \bar{\gamma}[\bot]$ or $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi \models \theta \wedge \bar{\gamma}[\bot]$ or $\pi^n \models \theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \theta \vee \bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. If $\pi \models \theta$ or $\pi^n \models \theta$, we trivially obtain $\pi \models \gamma[\bot]$ resp. $\pi^n \models \gamma[\varphi \wedge \psi]$. Otherwise, if $\pi \not\models \theta$ and $\pi^n \not\models \theta$, we know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^n \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi \models \bar{\gamma}[\bot]$ or $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]$, which trivially imply $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that $\pi^1 \models \bar{\gamma}[\varphi]$ and $\pi^{n+1} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^1 \models \bar{\gamma}[\bot]$ or $\pi^{n+1} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \mathbf{F}\,\bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or

$\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. So, if $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$, we trivially have $\pi \models \gamma[\bot]$, and if $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, we trivially have $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{G}\,\bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Thus, we have $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\theta$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. Thus, we have $\pi^{[0,k)} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l)} \models \bar{\gamma}[\psi]$. If $k < n$ and it does not hold that $\pi^{[0,k)} \models \bar{\gamma}[\bot]$ or $\pi^{[n,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, there must exist $i_0 < k$ and $j_0 < l$ such that $\pi^{i_0} \not\models \bar{\gamma}[\bot]$ and $\pi^{n+j_0} \not\models \bar{\gamma}[\varphi \wedge \psi]$. However, since $\pi^{i_0} \models \bar{\gamma}[\varphi]$ and $\pi^{n+j_0} \models \bar{\gamma}[\psi]$, this contradicts the induction hypothesis. Hence, we have either $\pi^{[0,k)} \models \bar{\gamma}[\bot]$ and $\pi^k \models \theta$ or $\pi^{[n,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+l} \models \theta$, that is, $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively. Otherwise, if $k \geq n$, we know that $\pi^{[n,\min(k,n+l))} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^{[n,\min(k,n+l))} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,\min(k,n+l))} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{\min(k,n+l)} \models \theta$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\theta = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta)$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. Thus, we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l]} \models \bar{\gamma}[\psi]$. If $k < n$ and it does not hold that $\pi^{[0,k]} \models \bar{\gamma}[\bot]$ or $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$, there must exist $i_0 \leq k$ and $j_0 \leq l$ such that $\pi^{i_0} \not\models \bar{\gamma}[\bot]$ and $\pi^{n+j_0} \not\models \bar{\gamma}[\varphi \wedge \psi]$. However, since $\pi^{i_0} \models \bar{\gamma}[\varphi]$ and $\pi^{n+j_0} \models \bar{\gamma}[\psi]$, this contradicts the induction hypothesis. Hence, we have either $\pi^{[0,k]} \models \bar{\gamma}[\bot]$ and $\pi^k \models \theta$ or $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+l} \models \theta$, that is, $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively. Otherwise, if $k \geq n$, we know that $\pi^{[n,\min(k,n+l)]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^{[n,\min(k,n+l)]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,\min(k,n+l)]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{\min(k,n+l)} \models \theta$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta\,\mathbf{U}\,\bar{\gamma}$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \bar{\gamma}[\bot]$, that is, $\pi \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$

implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\mathring{\mathbf{U}}\, \bar{\gamma} = \theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma})$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we have $\pi^{[0,k]} \models \theta$ and $\pi^k \models \bar{\gamma}[\bot]$, that is, $\pi \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\bar{\mathbf{U}}\, \bar{\gamma} = \theta \,\mathbf{U}\, (\neg\theta \wedge \bar{\gamma})$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\bot]$, that is, $\pi \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that $\pi^{[n,n+l)} \models \theta$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma} \,\mathbf{W}\, \theta = (\mathbf{G}\, \bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, \theta)$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between three cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that $\pi \models \mathbf{G}\, \bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\, \bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^n \models \mathbf{G}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, if there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{[n,\infty)} \not\models \theta$, we know that $k < n$ and $\pi^n \models \mathbf{G}\, \bar{\gamma}[\psi]$. Thus, we have $\pi^{[0,k)} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,\infty)} \models \bar{\gamma}[\psi]$. If it does not hold that $\pi^{[0,k)} \models \bar{\gamma}[\bot]$ or $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, there must exist $i_0 < k$ and j_0 such that $\pi^{i_0} \not\models \bar{\gamma}[\bot]$ and $\pi^{n+j_0} \not\models \bar{\gamma}[\varphi \wedge \psi]$. However, since $\pi^{i_0} \models \bar{\gamma}[\varphi]$ and $\pi^{n+j_0} \models \bar{\gamma}[\psi]$, this contradicts the induction hypothesis. Hence, we have either $\pi^{[0,k)} \models \bar{\gamma}[\bot]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\bot] \,\mathbf{U}\, \theta$, or $\pi^n \models \mathbf{G}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially imply $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively. (iii) Otherwise, there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. Thus, we have $\pi^{[0,k)} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l)} \models \bar{\gamma}[\psi]$. If $k < n$ and it does not hold that $\pi^{[0,k)} \models \bar{\gamma}[\bot]$ or $\pi^{[n,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, there must exist $i_0 < k$ and $j_0 < l$ such that $\pi^{i_0} \not\models \bar{\gamma}[\bot]$ and $\pi^{n+j_0} \not\models \bar{\gamma}[\varphi \wedge \psi]$. However, since $\pi^{i_0} \models \bar{\gamma}[\varphi]$ and $\pi^{n+j_0} \models \bar{\gamma}[\psi]$, this contradicts the induction hypothesis. Hence, we

have either $\pi^{[0,k)} \models \bar{\gamma}[\bot]$ and $\pi^k \models \theta$ or $\pi^{[n,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+l} \models \theta$, that is, $\pi \models \bar{\gamma}[\bot] \,\mathbf{U}\, \theta$ and $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{U}\, \theta$ respectively, which trivially imply $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively. Otherwise, if $k \geq n$, we know that $\pi^{[n,\min(k,n+l))} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^{[n,\min(k,n+l))} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,\min(k,n+l))} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{\min(k,n+l)} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{U}\, \theta$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \bar{\gamma} \,\hat{\mathbf{W}}\, \theta = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, (\bar{\gamma} \wedge \theta))$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between three cases: (i) If $\pi^{[0,\infty)} \not\models \theta$, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, if there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{[n,\infty)} \not\models \theta$, we know that $k < n$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$. Thus, we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,\infty)} \models \bar{\gamma}[\psi]$. If it does not hold that $\pi^{[0,k]} \models \bar{\gamma}[\bot]$ or $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, there must exist $i_0 \leq k$ and j_0 such that $\pi^{i_0} \not\models \bar{\gamma}[\bot]$ and $\pi^{n+j_0} \not\models \bar{\gamma}[\varphi \wedge \psi]$. However, since $\pi^{i_0} \models \bar{\gamma}[\varphi]$ and $\pi^{n+j_0} \models \bar{\gamma}[\psi]$, this contradicts the induction hypothesis. Hence, we have either $\pi^{[0,k]} \models \bar{\gamma}[\bot]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\bot] \,\mathring{\mathbf{U}}\, \theta$, or $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially imply $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively. (iii) Otherwise, there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. Thus, we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l]} \models \bar{\gamma}[\psi]$. If $k < n$ and it does not hold that $\pi^{[0,k]} \models \bar{\gamma}[\bot]$ or $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$, there must exist $i_0 \leq k$ and $j_0 \leq l$ such that $\pi^{i_0} \not\models \bar{\gamma}[\bot]$ and $\pi^{n+j_0} \not\models \bar{\gamma}[\varphi \wedge \psi]$. However, since $\pi^{i_0} \models \bar{\gamma}[\varphi]$ and $\pi^{n+j_0} \models \bar{\gamma}[\psi]$, this contradicts the induction hypothesis. Hence, we have either $\pi^{[0,k]} \models \bar{\gamma}[\bot]$ and $\pi^k \models \theta$ or $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+l} \models \theta$, that is, $\pi \models \bar{\gamma}[\bot] \,\mathring{\mathbf{U}}\, \theta$ and $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathring{\mathbf{U}}\, \theta$ respectively, which trivially imply $\pi \models \gamma[\bot]$ and $\pi^n \models \gamma[\varphi \wedge \psi]$ respectively. Otherwise, if $k \geq n$, we know that $\pi^{[n,\min(k,n+l)]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis and Lemma 2.1, we obtain $\pi^{[n,\min(k,n+l)]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,\min(k,n+l)]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{\min(k,n+l)} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathring{\mathbf{U}}\, \theta$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\mathbf{W}\, \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, \bar{\gamma})$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^n \models \mathbf{G}\,\theta$, we trivially obtain $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exist

least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \bar{\gamma}[\bot]$, that is, $\pi \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\mathring{\mathbf{W}}\, \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma}))$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^n \models \mathbf{G}\,\theta$, we trivially obtain $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we have $\pi^{[0,k]} \models \theta$ and $\pi^k \models \bar{\gamma}[\bot]$, that is, $\pi \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta \,\mathring{\mathbf{U}}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \,\bar{\mathbf{W}}\, \bar{\gamma} = (\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, (\neg\theta \wedge \bar{\gamma}))$. Suppose that $\pi \models \gamma[\varphi]$ and $\pi^n \models \gamma[\psi]$ for some $n \in \mathbb{N}$. Now, we have to distinguish between two cases: (i) If $\pi^n \models \mathbf{G}\,\theta$, we trivially obtain $\pi^n \models \gamma[\varphi \wedge \psi]$. (ii) Otherwise, there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\bot]$, that is, $\pi \models \gamma[\bot]$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that $\pi^{[n,n+l)} \models \theta$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta \,\bar{\mathbf{U}}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]$.

This concludes the proof. □

A.2 Proof of Maximality

Proof of Lemma 4.13

Lemma 4.13. *Let $\gamma \in LTLQ^1 \cup LTLQ^2$ be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi^{4n} \models \gamma[p]$, $\pi^{4n+2} \models \gamma[q]$, and $\pi^{2n} \not\models \gamma[p \wedge q]$ for all $n \in \mathbb{N}$.*

Proof. Structural induction on γ.

Induction start:

▷ If γ is the placeholder, the assertion holds for path π iff for all $n \in \mathbb{N}$ it holds that $p \in \ell(\pi^{4n})$, $q \in \ell(\pi^{4n+2})$, and $\{p, q\} \not\subseteq \ell(\pi^{2n})$.

Induction step:

▷ Let $\gamma = \alpha \wedge \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\neg\varphi$ implies $\neg(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \vee \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p]$, $\pi^{4n+2} \models \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since φ implies $\alpha \vee \varphi$ and $\neg\alpha \wedge \neg\varphi$ is equivalent to $\neg(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. By induction hypothesis, we obtain a path σ for $\bar{\gamma}$. Let $\pi = s \circ \sigma$, where s is any state. So we have $\pi^{4n} \models \mathbf{X}\,\bar{\gamma}[p]$, $\pi^{4n+2} \models \mathbf{X}\,\bar{\gamma}[q]$, and $\pi^{2n} \models \mathbf{X}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{X}\,\neg\varphi$ is equivalent to $\neg\mathbf{X}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\alpha$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ and $\alpha \in \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \mathbf{X}\,\alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \mathbf{X}\,\alpha$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q] \wedge \neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \mathbf{X}\,\alpha$ implies $\varphi\,\mathbf{U}\,\alpha$ and $\neg\varphi \wedge \neg\alpha$ implies $\neg(\varphi\,\mathbf{U}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\alpha = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha)$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{U}}\,\alpha$ and $\neg\varphi$ implies $\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathbf{U}\, \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p]$, $\pi^{4n+2} \models \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since φ implies $\alpha \,\mathbf{U}\, \varphi$ and $\neg\alpha \wedge \neg\varphi$ implies $\neg(\alpha \,\mathbf{U}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathring{\mathbf{U}}\, \bar{\gamma} = \alpha \,\mathbf{U}\, (\alpha \wedge \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q] \wedge \mathbf{X}\neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha \,\mathring{\mathbf{U}}\, \varphi$ and $\neg\varphi \wedge \mathbf{X}\neg\alpha$ implies $\neg(\alpha \,\mathring{\mathbf{U}}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\bar{\mathbf{U}}\, \bar{\gamma} = \alpha \,\mathbf{U}\, (\neg\alpha \wedge \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \neg\alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \neg\alpha \wedge \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\neg\alpha \wedge \varphi$ implies $\alpha \,\bar{\mathbf{U}}\, \varphi$ and $\neg\alpha \wedge \neg\varphi$ implies $\neg(\alpha \,\bar{\mathbf{U}}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma} \,\mathbf{W}\, \alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, \alpha)$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ and $\alpha \in \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \mathbf{X}\,\alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \mathbf{X}\,\alpha$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q] \wedge \neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \mathbf{X}\,\alpha$ implies $\varphi \,\mathbf{W}\, \alpha$ and $\neg\varphi \wedge \neg\alpha$ implies $\neg(\varphi \,\mathbf{W}\, \alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma} \,\mathring{\mathbf{W}}\, \alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, (\bar{\gamma} \wedge \alpha))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi \,\mathring{\mathbf{W}}\, \alpha$ and $\neg\varphi$ implies $\neg(\varphi \,\mathring{\mathbf{W}}\, \alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathbf{W}\, \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \,\mathbf{U}\, \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p]$, $\pi^{4n+2} \models \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since φ implies $\alpha \,\mathbf{W}\, \varphi$ and $\neg\alpha \wedge \neg\varphi$ implies $\neg(\alpha \,\mathbf{W}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \mathring{\mathbf{W}} \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \mathbf{U} (\alpha \wedge \bar{\gamma}))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\bar{\gamma}[p \wedge q] \wedge \mathbf{X}\neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha \mathring{\mathbf{W}} \varphi$ and $\neg\varphi \wedge \mathbf{X}\neg\alpha$ implies $\neg(\alpha \mathring{\mathbf{W}} \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \bar{\mathbf{W}} \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \mathbf{U} (\neg\alpha \wedge \bar{\gamma}))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \neg\alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \neg\alpha \wedge \bar{\gamma}[q]$, and $\pi^{2n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\neg\alpha \wedge \varphi$ implies $\alpha \bar{\mathbf{W}} \varphi$ and $\neg\alpha \wedge \neg\varphi$ implies $\neg(\alpha \bar{\mathbf{W}} \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

This concludes the proof. □

Proof of Lemma 4.14

Lemma 4.14. *Let $\gamma \in$ LTLQ^1 be simple. Further, let p and q be atomic propositions not occurring in γ. Then, there exists a path π such that $\pi^{4n} \models \gamma[p]$ and $\pi^{4n+2} \models \gamma[q]$ for all $n \in \mathbb{N}$ as well as $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ.

Induction start:

▷ If γ is the placeholder, the assertion holds for path π iff for all $n \in \mathbb{N}$ it holds that $p \in \ell(\pi^{4n})$, $q \in \ell(\pi^{4n+2})$, and $\{p, q\} \not\subseteq \ell(\pi^n)$.

▷ Let $\gamma = \alpha \wedge \bar{\gamma}$ such that $\bar{\gamma} \in$ LTLQ^2. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,(\neg\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma} \mathring{\mathbf{U}} \alpha = \bar{\gamma} \mathbf{U} (\bar{\gamma} \wedge \alpha)$ such that $\bar{\gamma} \in$ LTLQ^2. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$,

$\pi^{4n+2} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi \models \mathbf{G}\,(\neg\bar{\gamma}[p \wedge q] \vee \neg\alpha)$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{U}}\,\alpha$ and $\mathbf{G}\,(\neg\varphi \vee \neg\alpha)$ is equivalent to $\mathbf{G}\,\neg(\varphi \wedge \alpha)$ which implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathring{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\alpha \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^2$. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha\,\mathring{\mathbf{U}}\,\varphi$ and $\mathbf{G}\,(\neg\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha\,\mathring{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^2$. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ and $\alpha \in \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \neg\alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \neg\alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\neg\alpha \wedge \varphi$ implies $\alpha\,\bar{\mathbf{U}}\,\varphi$ and $\mathbf{G}\,(\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\neg\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{W}}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^2$. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi \models \mathbf{GF}\,\neg\bar{\gamma}[p \wedge q] \wedge \mathbf{G}\,(\neg\bar{\gamma}[p \wedge q] \vee \neg\alpha)$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{W}}\,\alpha$ and $\mathbf{GF}\,\neg\varphi \wedge \mathbf{G}\,(\neg\varphi \vee \neg\alpha)$ is equivalent to $\mathbf{GF}\,\neg\varphi \wedge \mathbf{G}\,\neg(\varphi \wedge \alpha)$ which implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathring{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,(\alpha \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^2$. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ and $\alpha \notin \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\neg\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha\,\mathring{\mathbf{W}}\,\varphi$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\neg\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg(\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha\,\mathring{\mathbf{W}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^2$. By Lemma 4.13, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ and $\alpha \in \ell(\pi^{2n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \neg\alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \neg\alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\alpha \vee \neg\bar{\gamma}[p \wedge q])$

for all $n \in \mathbb{N}$. Thus, since $\neg\alpha \wedge \varphi$ implies $\alpha \,\bar{\mathbf{W}}\, \varphi$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg(\neg\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha \,\bar{\mathbf{W}}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

Induction step:

▷ Let $\gamma = \alpha \wedge \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \vee \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p]$, $\pi^{4n+2} \models \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \wedge \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since φ implies $\alpha \vee \varphi$ and $\mathbf{G}\,(\neg\alpha \wedge \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. By induction hypothesis, we obtain a path σ for $\bar{\gamma}$. Let $\pi = s \circ \sigma$, where s is any state. So we have $\pi^{4n} \models \mathbf{X}\,\bar{\gamma}[p]$, $\pi^{4n+2} \models \mathbf{X}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{XG}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{XG}\,\neg\varphi$ is equivalent to $\mathbf{G}\,\neg\mathbf{X}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\alpha = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha)$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{U}}\,\alpha$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \neg\alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \neg\alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\neg\alpha \wedge \varphi$ implies $\alpha\,\bar{\mathbf{U}}\,\varphi$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\bar{\mathbf{W}}}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{2n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n+2} \models \bar{\gamma}[q] \wedge \alpha$,

and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{W}}\,\alpha$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma}))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \neg\alpha \wedge \bar{\gamma}[p]$, $\pi^{4n+2} \models \neg\alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\neg\alpha \wedge \varphi$ implies $\alpha\,\bar{\mathbf{W}}\,\varphi$ and $\mathbf{G}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{W}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

This concludes the proof. □

Proof of Lemma 4.15

Lemma 4.15. *Let $\gamma \in LTLQ^5 \cup LTLQ^6$ be simple. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^3$ and $LTLQ^4$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Then, there exists a path π such that $\pi^{4n} \models \gamma[p] \wedge \gamma[q]$ for all $n \in \mathbb{N}$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \alpha \wedge \bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ and $\alpha \notin \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,(\neg\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathring{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\alpha \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ and $\alpha \notin \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha\,\mathring{\mathbf{U}}\,\varphi$ and $\mathbf{G}\,(\neg\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha\,\mathring{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathbf{W}\, \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \,\mathbf{U}\, \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1$. By Lemma 4.14, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{[4n,4n+2)})$ and $\alpha \notin \ell(\pi^{4n+2})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \mathbf{X}\,\alpha \wedge \mathbf{XX}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since both φ and $\alpha \wedge \mathbf{X}\,\alpha \wedge \mathbf{XX}\,\varphi$ imply $\alpha \,\mathbf{W}\, \varphi$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha \,\mathbf{W}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathring{\mathbf{W}}\, \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \,\mathbf{U}\, (\alpha \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1$. By Lemma 4.14, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{[4n,4n+2]})$ and $\alpha \notin \ell(\pi^{4n+3})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \mathbf{X}\,\alpha \wedge \mathbf{XX}\,(\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since both $\alpha \wedge \varphi$ and $\alpha \wedge \mathbf{X}\,\alpha \wedge \mathbf{XX}\,(\alpha \wedge \varphi)$ imply $\alpha \,\mathring{\mathbf{W}}\, \varphi$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha \,\mathring{\mathbf{W}}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathring{\mathbf{W}}\, \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \,\mathbf{U}\, (\alpha \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^3 \cup \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ and $\alpha \notin \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\neg\alpha \vee \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha \,\mathring{\mathbf{W}}\, \varphi$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\neg\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg(\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha \,\mathring{\mathbf{W}}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

Induction step:

▷ Let $\gamma = \alpha \wedge \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \vee \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p]$, $\pi^{4n} \models \bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \wedge \neg\bar{\gamma}[p \wedge q])$ for all $n \in \mathbb{N}$. Thus, since φ implies $\alpha \vee \varphi$ and $\mathbf{G}\,(\neg\alpha \wedge \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. By induction hypothesis, we obtain a path σ for $\bar{\gamma}$. Let $\pi = s \circ \sigma$, where s is any state. So we have $\pi^{4n} \models \mathbf{X}\,\bar{\gamma}[p]$, $\pi^{4n} \models \mathbf{X}\,\bar{\gamma}[q]$,

and $\pi \models \mathbf{XG}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{XG}\,\neg\varphi$ is equivalent to $\mathbf{G}\,\neg\mathbf{X}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

- ▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\alpha = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha)$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{U}}\,\alpha$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

- ▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{W}}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \bar{\gamma}[p] \wedge \alpha$, $\pi^{4n} \models \bar{\gamma}[q] \wedge \alpha$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\varphi \wedge \alpha$ implies $\varphi\,\mathring{\mathbf{W}}\,\alpha$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

- ▷ Let $\gamma = \alpha\,\mathring{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,(\alpha \wedge \bar{\gamma}))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ and $\alpha \notin \ell(\pi^{4n+1})$ for all $n \in \mathbb{N}$. So we have $\pi^{4n} \models \alpha \wedge \bar{\gamma}[p]$, $\pi^{4n} \models \alpha \wedge \bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\alpha \wedge \varphi$ implies $\alpha\,\mathring{\mathbf{W}}\,\varphi$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathring{\mathbf{W}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

This concludes the proof. □

Proof of Lemma 4.16

Lemma 4.16. *Let $\gamma = \mathbf{G}\,\bar{\bar{\gamma}}$ be a simple LTL query where $\bar{\bar{\gamma}} \in LTLQ^6$. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^4$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma \models \mathbf{G}\,\bar{\gamma}[p] \wedge \mathbf{G}\,\bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Suppose further that for every $LTLQ^5$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Then, there exists a path π such that $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \mathbf{G}(\alpha \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}(\alpha \wedge \bar{\gamma}[p])$, $\pi \models \mathbf{G}(\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{GF}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}(\alpha \wedge \varphi)$ is equivalent to $\mathbf{GG}(\alpha \wedge \varphi)$ and $\mathbf{GF}\,\neg\varphi$ implies $\mathbf{GF}\,\neg(\alpha \wedge \varphi)$ which is equivalent to $\mathbf{G}\,\neg\mathbf{G}(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}(\alpha \vee \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^5$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}(\alpha \vee \bar{\gamma}[p])$, $\pi \models \mathbf{G}(\alpha \vee \bar{\gamma}[q])$, and $\pi \models \mathbf{GF}(\neg\alpha \wedge \neg\bar{\gamma}[p \wedge q])$. Thus, since $\mathbf{G}(\alpha \vee \varphi)$ is equivalent to $\mathbf{GG}(\alpha \vee \varphi)$ and $\mathbf{GF}(\neg\alpha \wedge \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg\mathbf{G}(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

Induction step:

▷ Let $\gamma = \mathbf{G}(\alpha \wedge \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\mathbf{G}\,\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}(\alpha \wedge \bar{\gamma}[p])$, $\pi \models \mathbf{G}(\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}(\alpha \wedge \varphi)$ is equivalent to $\mathbf{GG}(\alpha \wedge \varphi)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{GF}\,\neg(\alpha \wedge \varphi)$ which is equivalent to $\mathbf{G}\,\neg\mathbf{G}(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}(\alpha \vee \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\mathbf{G}\,\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}(\neg\alpha \wedge \neg\bar{\gamma}[p \wedge q])$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{GG}(\alpha \vee \varphi)$ and $\mathbf{G}(\neg\alpha \wedge \neg\varphi)$ implies $\mathbf{GF}(\neg\alpha \wedge \neg\varphi)$ which is equivalent to $\mathbf{G}\,\neg\mathbf{G}(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{GX}\,\bar{\gamma}$. By induction hypothesis, we obtain a path σ for $\mathbf{G}\,\bar{\gamma}$. Let $\pi = s \circ \sigma$, where s is any state. So we have $\pi \models \mathbf{XG}\,\bar{\gamma}[p]$, $\pi \models \mathbf{XG}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{XG}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{XG}\,\varphi$ is equivalent to $\mathbf{GGX}\,\varphi$ and $\mathbf{XG}\,\neg\varphi$ implies $\mathbf{XGF}\,\neg\varphi$ which is equivalent to $\mathbf{G}\,\neg\mathbf{GX}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\alpha) = \mathbf{G}(\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha))$. By induction hypothesis, we obtain a path π for $\mathbf{G}\,\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}(\bar{\gamma}[p] \wedge \alpha)$, $\pi \models \mathbf{G}(\bar{\gamma}[q] \wedge \alpha)$,

and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\varphi \wedge \alpha)$ implies $\mathbf{GG}\,(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{GF}\,\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ which is equivalent to $\mathbf{G}\,\neg\mathbf{G}\,(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}\,(\bar{\gamma}\,\mathring{\mathbf{W}}\,\alpha) = \mathbf{G}\,((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha)))$. By induction hypothesis, we obtain a path π for $\mathbf{G}\,\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\bar{\gamma}[p] \wedge \alpha)$, $\pi \models \mathbf{G}\,(\bar{\gamma}[q] \wedge \alpha)$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\varphi \wedge \alpha)$ implies $\mathbf{GG}\,(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{GF}\,\neg(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ which is equivalent to $\mathbf{G}\,\neg\mathbf{G}\,(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

This concludes the proof. □

Proof of Lemma 4.17

Lemma 4.17. *Let $\gamma \in LTLQ^3$ be simple. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^4$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma \models \mathbf{G}\,\bar{\gamma}[p] \wedge \mathbf{G}\,\bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Suppose further that for every $LTLQ^5$ and $LTLQ^6$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ for all $n \in \mathbb{N}$ and $\sigma \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Then, there exists a path π such that $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ and $\pi \models \mathbf{G}\,\neg\gamma[p \wedge q]$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \mathbf{F}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By Lemma 4.14 resp. assumption, we obtain a path π for $\bar{\gamma}$. So we have $\pi \models \mathbf{GF}\,\bar{\gamma}[p]$, $\pi \models \mathbf{GF}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\neg\varphi$ is equivalent to $\mathbf{G}\,\neg\mathbf{F}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\varphi$ is equivalent to $\mathbf{GG}\,\varphi$ and $\mathbf{GF}\,\neg\varphi$ is equivalent to $\mathbf{G}\,\neg\mathbf{G}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^6$. Then, we obtain the assertion by Lemma 4.16.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\alpha = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha)$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^{4n})$ and $\alpha \notin \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p] \wedge \mathbf{GF}\,\alpha$, $\pi \models \mathbf{G}\,\bar{\gamma}[q] \wedge \mathbf{GF}\,\alpha$, and $\pi \models \mathbf{G}\,(\neg\bar{\gamma}[p \wedge q] \vee \neg\alpha)$. Thus, since $\mathbf{G}\,\varphi \wedge \mathbf{GF}\,\alpha$ implies $\mathbf{G}\,(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ and $\mathbf{G}\,(\neg\varphi \vee \neg\alpha)$ is equivalent $\mathbf{G}\,\neg(\varphi \wedge \alpha)$ which implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathbf{U}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By Lemma 4.14 resp. assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\alpha \wedge \mathbf{GF}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\alpha \wedge \mathbf{GF}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\alpha \wedge \mathbf{GF}\,\varphi$ implies $\mathbf{G}\,(\alpha\,\mathbf{U}\,\varphi)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathbf{U}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathring{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\alpha \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^1 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By Lemma 4.14 resp. assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\alpha \wedge \mathbf{GF}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\alpha \wedge \mathbf{GF}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\alpha \wedge \mathbf{GF}\,\varphi$ implies $\mathbf{G}\,(\alpha\,\mathring{\mathbf{U}}\,\varphi)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathring{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\alpha \vee \bar{\gamma}[p]) \wedge \mathbf{GF}\,\neg\alpha$, $\pi \models \mathbf{G}\,(\alpha \vee \bar{\gamma}[q]) \wedge \mathbf{GF}\,\neg\alpha$, and $\pi \models \mathbf{G}\,(\alpha \vee \neg\bar{\gamma}[p \wedge q])$. Thus, since $\mathbf{G}\,(\alpha \vee \varphi) \wedge \mathbf{GF}\,\neg\alpha$ implies $\mathbf{G}\,(\alpha\,\bar{\mathbf{U}}\,\varphi)$ and $\mathbf{G}\,(\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\neg\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\alpha)$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\bar{\gamma}[p \wedge q] \wedge \mathbf{G}\,\neg\alpha$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\varphi\,\mathbf{W}\,\alpha)$ and $\mathbf{GF}\,\neg\varphi \wedge \mathbf{G}\,\neg\alpha$ implies $\mathbf{G}\,\neg(\varphi\,\mathbf{W}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\alpha)$ such that $\bar{\gamma} \in \mathrm{LTLQ}^6$. By Lemma 4.16, we obtain a path π for $\mathbf{G}\,\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g.

that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{GG}\,\bar{\gamma}[p]$, $\pi \models \mathbf{GG}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\mathbf{G}\,\bar{\gamma}[p \wedge q] \wedge \mathbf{G}\,\neg\alpha$. Thus, since $\mathbf{GG}\,\varphi$ implies $\mathbf{G}\,(\varphi\,\mathbf{W}\,\alpha)$ and $\mathbf{G}\,\neg\mathbf{G}\,\varphi \wedge \mathbf{G}\,\neg\alpha$ implies $\mathbf{G}\,\neg(\varphi\,\mathbf{W}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{W}}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{GF}\,\neg\bar{\gamma}[p \wedge q] \wedge \mathbf{G}\,\neg\alpha$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ and $\mathbf{GF}\,\neg\varphi \wedge \mathbf{G}\,\neg\alpha$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathbf{W}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,\bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\alpha \vee \bar{\gamma}[p])$, $\pi \models \mathbf{G}\,(\alpha \vee \bar{\gamma}[q])$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\alpha \vee \varphi)$ implies $\mathbf{G}\,(\alpha\,\mathbf{W}\,\varphi)$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathbf{W}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{LTLQ}^4 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\alpha \vee \bar{\gamma}[p])$, $\pi \models \mathbf{G}\,(\alpha \vee \bar{\gamma}[q])$, and $\pi \models \mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\alpha \vee \neg\bar{\gamma}[p \wedge q])$. Thus, since $\mathbf{G}\,(\alpha \vee \varphi)$ implies $\mathbf{G}\,(\alpha\,\bar{\mathbf{W}}\,\varphi)$ and $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,(\alpha \vee \neg\varphi)$ is equivalent to $\mathbf{GF}\,\neg\alpha \wedge \mathbf{G}\,\neg(\neg\alpha \wedge \varphi)$ which implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{W}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

Induction step:

▷ Let $\gamma = \alpha \wedge \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\alpha \wedge \bar{\gamma}[p])$, $\pi \models \mathbf{G}\,(\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha \wedge \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \vee \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,(\neg\alpha \wedge \neg\bar{\gamma}[p \wedge q])$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\alpha \vee \varphi)$ and $\mathbf{G}\,(\neg\alpha \wedge \neg\varphi)$ is equivalent to $\mathbf{G}\,\neg(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. By induction hypothesis, we obtain a path σ for $\bar{\gamma}$. Let $\pi = s \circ \sigma$, where s is any state. So we have $\pi \models \mathbf{XG}\,\bar{\gamma}[p]$, $\pi \models \mathbf{XG}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{XG}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{XG}\,\varphi$ is equivalent to $\mathbf{GX}\,\varphi$ and $\mathbf{XG}\,\neg\varphi$ is equivalent to $\mathbf{G}\,\neg\mathbf{X}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{F}\,\bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{GF}\,\varphi$ and $\mathbf{G}\,\neg\varphi$ is equivalent to $\mathbf{G}\,\neg\mathbf{F}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{G}\,\bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\varphi$ is equivalent to $\mathbf{GG}\,\varphi$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{GF}\,\neg\varphi$ which is equivalent to $\mathbf{G}\,\neg\mathbf{G}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{U}}\,\alpha = \bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha)$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\bar{\gamma}[p] \wedge \alpha)$, $\pi \models \mathbf{G}\,(\bar{\gamma}[q] \wedge \alpha)$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\varphi \wedge \alpha)$ implies $\mathbf{G}\,(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{U}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathbf{U}\,\bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\alpha\,\mathbf{U}\,\varphi)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathbf{U}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathring{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\alpha \wedge \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \in \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\alpha \wedge \bar{\gamma}[p])$, $\pi \models \mathbf{G}\,(\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\alpha \wedge \varphi)$ implies $\mathbf{G}\,(\alpha\,\mathring{\mathbf{U}}\,\varphi)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathring{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{U}}\,\bar{\gamma} = \alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\neg\alpha \wedge \bar{\gamma}[p])$, $\pi \models \mathbf{G}\,(\neg\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\neg\alpha \wedge \varphi)$ implies $\mathbf{G}\,(\alpha\,\bar{\mathbf{U}}\,\varphi)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{U}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\alpha)$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q] \wedge \mathbf{G}\,\neg\alpha$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\varphi\,\mathbf{W}\,\alpha)$ and $\mathbf{G}\,\neg\varphi \wedge \mathbf{G}\,\neg\alpha$ implies $\mathbf{G}\,\neg(\varphi\,\mathbf{W}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathring{\mathbf{W}}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \alpha))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ and $\mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\varphi\,\mathring{\mathbf{W}}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathbf{W}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,\bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi \models \mathbf{G}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\alpha\,\mathbf{W}\,\varphi)$ and $\mathbf{G}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\mathbf{W}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\bar{\mathbf{W}}\,\bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha\,\mathbf{U}\,(\neg\alpha \wedge \bar{\gamma}))$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^n)$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\neg\alpha \wedge \bar{\gamma}[p])$, $\pi \models \mathbf{G}\,(\neg\alpha \wedge \bar{\gamma}[q])$, and $\pi \models \mathbf{G}\,\neg\alpha \wedge \mathbf{G}\,\neg\bar{\gamma}[p \wedge q]$. Thus, since $\mathbf{G}\,(\neg\alpha \wedge \varphi)$ implies $\mathbf{G}\,(\alpha\,\bar{\mathbf{W}}\,\varphi)$ and $\mathbf{G}\,\neg\alpha \wedge \mathbf{G}\,\neg\varphi$ implies $\mathbf{G}\,\neg(\alpha\,\bar{\mathbf{W}}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

This concludes the proof. □

Proof of Lemma 4.18

Lemma 4.18. *Let $\gamma \in LTLQ^4$ be simple. Further, let p and q be atomic propositions not occurring in γ. Suppose that for every $LTLQ^3$, $LTLQ^5$, and $LTLQ^6$ subquery $\bar{\gamma}$ there exists a path σ such that $\sigma^{4n} \models \bar{\gamma}[p] \wedge \bar{\gamma}[q]$ and $\sigma^{4n} \not\models \bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Then, there exists a path π such that $\pi \models \mathbf{G}\,\gamma[p] \wedge \mathbf{G}\,\gamma[q]$ and $\pi^{4n} \not\models \gamma[p \wedge q]$ for all $n \in \mathbb{N}$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\alpha$ where $\bar{\gamma} \in \mathrm{LTLQ}^3 \cup \mathrm{LTLQ}^5 \cup \mathrm{LTLQ}^6$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g.

that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\bar{\gamma}[p] \vee \alpha) \wedge \mathbf{GF}\,\alpha$, $\pi \models \mathbf{G}\,(\bar{\gamma}[q] \vee \alpha) \wedge \mathbf{GF}\,\alpha$, and $\pi^{4n} \models \neg\bar{\gamma}[p \wedge q] \wedge \neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,(\varphi \vee \alpha) \wedge \mathbf{GF}\,\alpha$ implies $\mathbf{G}\,(\varphi\,\mathbf{U}\,\alpha)$ and $\neg\varphi \wedge \neg\alpha$ implies $\neg(\varphi\,\mathbf{U}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{W}\,\alpha = (\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\alpha)$ where $\bar{\gamma} \in \mathrm{LTLQ}^5$. By assumption, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{[4n+1,4n+3]})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,(\bar{\gamma}[p] \vee \alpha)$, $\pi \models \mathbf{G}\,(\bar{\gamma}[q] \vee \alpha)$, and $\pi^{4n} \models \neg\bar{\gamma}[p \wedge q] \wedge \neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,(\varphi \vee \alpha)$ implies $\mathbf{G}\,(\varphi\,\mathbf{W}\,\alpha)$ and $\neg\varphi \wedge \neg\alpha$ implies $\neg(\varphi\,\mathbf{W}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

Induction step:

▷ Let $\gamma = \alpha \vee \bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi^{4n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\alpha \vee \varphi)$ and $\neg\alpha \wedge \neg\varphi$ is equivalent to $\neg(\alpha \vee \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \mathbf{X}\,\bar{\gamma}$. By induction hypothesis, we obtain a path σ for $\bar{\gamma}$. Let $\pi = s \circ \sigma$, where s is any state. So we have $\pi \models \mathbf{XG}\,\bar{\gamma}[p]$, $\pi \models \mathbf{XG}\,\bar{\gamma}[q]$, and $\pi^{4n} \models \mathbf{X}\,\neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{XG}\,\varphi$ is equivalent to $\mathbf{GX}\,\varphi$ and $\mathbf{X}\,\neg\varphi$ is equivalent to $\neg\mathbf{X}\,\varphi$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \bar{\gamma}\,\mathbf{U}\,\alpha$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ and $\alpha \in \ell(\pi^{4n+1})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p] \wedge \mathbf{GF}\,\alpha$, $\pi \models \mathbf{G}\,\bar{\gamma}[q] \wedge \mathbf{GF}\,\alpha$, and $\pi^{4n} \models \neg\bar{\gamma}[p \wedge q] \wedge \neg\alpha$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,\varphi \wedge \mathbf{GF}\,\alpha$ implies $\mathbf{G}\,(\varphi\,\mathbf{U}\,\alpha)$ and $\neg\varphi \wedge \neg\alpha$ implies $\neg(\varphi\,\mathbf{U}\,\alpha)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha\,\mathbf{U}\,\bar{\gamma}$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi^{4n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\alpha\,\mathbf{U}\,\varphi)$ and $\neg\alpha \wedge \neg\varphi$ implies $\neg(\alpha\,\mathbf{U}\,\varphi)$ for all formulas φ, we obtain the assertion for γ on π.

▷ Let $\gamma = \alpha \,\mathbf{W}\, \bar{\gamma} = (\mathbf{G}\,\alpha) \vee (\alpha \,\mathbf{U}\, \bar{\gamma})$. By induction hypothesis, we obtain a path π for $\bar{\gamma}$. Since $\alpha \notin \mathrm{aprop}(\bar{\gamma})$, we can assume w.l.o.g. that $\alpha \notin \ell(\pi^{4n})$ for all $n \in \mathbb{N}$. So we have $\pi \models \mathbf{G}\,\bar{\gamma}[p]$, $\pi \models \mathbf{G}\,\bar{\gamma}[q]$, and $\pi^{4n} \models \neg\alpha \wedge \neg\bar{\gamma}[p \wedge q]$ for all $n \in \mathbb{N}$. Thus, since $\mathbf{G}\,\varphi$ implies $\mathbf{G}\,(\alpha \,\mathbf{W}\, \varphi)$ and $\neg\alpha \wedge \neg\varphi$ implies $\neg(\alpha \,\mathbf{W}\, \varphi)$ for all formulas φ, we obtain the assertion for γ on π.

This concludes the proof. □

Appendix B

Omitted Proofs for CTLQ$^{\mathcal{X}}$

B.1 Proof of Exactness

Proof of Lemma 4.23

Lemma 4.23. *Let $\gamma \in CTLQ^6 \cup CTLQ^7$. Suppose that every subquery in $CTLQ^8$ and $CTLQ^9$ is strong collecting. Then, γ is boundary collecting.*

Proof. Structural induction on γ.
Let s_1 and s_2 be any states in a Kripke structure such that $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\rho \in \text{paths}(s_1)$ such that $\rho(n) = s_2$ for some $n \in \mathbb{N}$. Let us define the set $\Pi_{s_2}^{s_1} = \{\pi \in \text{paths}(s_1) \mid \forall i \leq n.\ \pi(i) = \rho(i)\}$. It is easy to see that $\Pi_{s_2}^{s_1}$ is not empty and that $\text{paths}(s_2) = \{\pi^n \mid \pi \in \Pi_{s_2}^{s_1}\}$.

Induction start:

- ▷ Let $\gamma = \theta \vee \bar{\gamma}$ such that $\bar{\gamma} \in \text{CTLQ}^8 \cup \text{CTLQ}^9$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. If $s_1 \models \theta$ or $s_2 \models \theta$, we trivially obtain $s_1 \models \gamma[\bot]$ resp. $s_2 \models \gamma[\varphi \wedge \psi]$. Otherwise, we know that $s_1 \models \bar{\gamma}[\varphi]$ and $s_2 \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $s_2 \models \gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$ such that $\bar{\gamma} \in \text{CTLQ}^8 \cup \text{CTLQ}^9$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. If $s_1 \models \theta$, we trivially obtain $s_1 \models \gamma[\bot]$. Otherwise, we choose w.l.o.g. any path $\pi \in \Pi_{s_2}^{s_1}$. Recall that by definition we have $\pi^n \in \text{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that $\pi \models \bar{\gamma}[\varphi]$ and that there exists a least $k \in \mathbb{N}$ such that $\pi^{n+k} \models \theta$ and therefore $\pi^{[n,n+k)} \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[n,n+k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,n+k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+k} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{U}\,\theta$. Thus, since $\pi \in \Pi_{s_2}^{s_1}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\theta))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. If $s_1 \models \theta$, we trivially obtain $s_1 \models \gamma[\bot]$. Otherwise, we choose w.l.o.g. any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exists a least $k \in \mathbb{N}$ such that $\pi^{n+k} \models \theta$, we know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+k)} \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[n,n+k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,n+k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+k} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{U}\,\theta$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{W}\,\theta$. (ii) Otherwise, if no such k exists, we know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^{[n,\infty)} \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{W}\,\theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. Then, we know that $s_1 \models \bar{\gamma}[\varphi]$ and $s_2 \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $s_1 \models \bar{\gamma}[\bot]$ or $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, we have $s_1 \models \theta$ and $s_2 \models \theta$ as well as $s_1 \models \bar{\gamma}[\bot]$ or $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$, which imply $s_1 \models \gamma[\bot]$ and $s_2 \models \gamma[\varphi \wedge \psi]$ respectively.

▷ Let $\gamma = \theta \vee \bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. If $s_1 \models \theta$ or $s_2 \models \theta$, we trivially obtain $s_1 \models \gamma[\bot]$ resp. $s_2 \models \gamma[\varphi \wedge \psi]$. Otherwise, we know that $s_1 \models \bar{\gamma}[\varphi]$ and $s_2 \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $s_1 \models \bar{\gamma}[\bot]$ or $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$, which trivially imply $s_1 \models \gamma[\bot]$ and $s_2 \models \gamma[\varphi \wedge \psi]$ respectively.

This concludes the proof. □

Proof of Lemma 4.24

Lemma 4.24. *Let $\gamma \in CTLQ^1 \cup CTLQ^2$. Suppose that every subquery in $CTLQ^3$, $CTLQ^4$, and $CTLQ^5$ is intermediate collecting, and every subquery in $CTLQ^8$ and $CTLQ^9$ is strong collecting. Then, γ is weak collecting.*

Proof. Structural induction on γ.

Induction start:

▷ If γ is the placeholder, the assertion follows trivially.

▷ Let $\gamma = \theta \wedge \bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Since $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\theta \wedge (\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$, we obtain by assumption $\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\gamma[\varphi \wedge \psi]$. Hence, γ is weak collecting.

▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5 \cup \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that $\pi^1 \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption resp. Lemma 4.23, we obtain $\pi^1 \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \mathbf{X}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta) = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and therefore $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{U}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{U}\,(\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^l \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k,l) \leq r < \max(k,l)$ and $\pi^r \models \bar{\gamma}[\bot]$. Consequently, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we have either $\pi^{[0,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[0,\max(k,l)]} \models \theta$ and $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta\,\mathring{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{U}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{U}\,(\neg\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta\,\bar{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{W}}\,\theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$, we know that $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$, which trivially implies $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{W}}\,\theta$. (ii) Otherwise, if no such k exists, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[0,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{W}}\,\theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\mathring{\mathbf{U}}\,\bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If $\pi \models \mathbf{G}\,\theta$, we trivially have $\pi \models \theta\,\mathring{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^l \models \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k,l) \leq r < \max(k,l)$ and $\pi^r \models \bar{\gamma}[\bot]$. Consequently, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we have either $\pi^{[0,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[0,\max(k,l)]} \models \theta$ and $\pi^{\max(k,l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta\,\mathring{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \theta\,\mathring{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\bar{\mathbf{U}}\,\bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If $\pi \models \mathbf{G}\,\theta$, we trivially have $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta\,\bar{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. Since $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\theta \wedge (\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$, we obtain by induction hypothesis $\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \theta \vee \bar{\gamma}$. Since $\gamma[\varphi] \wedge \gamma[\psi]$ is equivalent to $\theta \vee (\bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi])$, we obtain by induction hypothesis $\theta \vee \bar{\gamma}[\varphi \wedge \psi]$, that is, $\gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that $\pi^1 \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^1 \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \mathbf{X}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta) = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta))$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$ and therefore $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{U}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{U}\,(\neg\theta \wedge \bar{\gamma}))$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta\,\bar{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{W}}\,\theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta))$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$, we know that $\pi^{[0,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$, which trivially implies $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{W}}\,\theta$. (ii) Otherwise, if no such k exists, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[0,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{W}}\,\theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\bar{\mathbf{U}}\,\bar{\gamma}))$. Suppose that $s \models \gamma[\varphi] \wedge \gamma[\psi]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If $\pi \models \mathbf{G}\,\theta$, we trivially have $\pi \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \neg\theta$ and therefore $\pi^k \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain

$\pi^k \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[0,k)} \models \theta$ and $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi \models \theta \,\bar{\mathbf{U}}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi \models \theta \,\bar{\mathbf{W}}\, \bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \text{paths}(s)$ was chosen w.l.o.g., we obtain $s \models \gamma[\varphi \wedge \psi]$.

This concludes the proof. □

Proof of Lemma 4.25

Lemma 4.25. *Let $\gamma \in CTLQ^3 \cup CTLQ^4 \cup CTLQ^5$. Suppose that every subquery in $CTLQ^1$ and $CTLQ^2$ is weak collecting, and every subquery in $CTLQ^8$ and $CTLQ^9$ is strong collecting. Then, γ is intermediate collecting.*

Proof. Structural induction on γ.
Let s_1 and s_2 be any states in a Kripke structure such that $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\rho \in \text{paths}(s_1)$ such that $\rho(n) = s_2$ for some $n \in \mathbb{N}$. Let us define the set $\Pi^{s_1}_{s_2} = \{\pi \in \text{paths}(s_1) \mid \forall i \leq n.\ \pi(i) = \rho(i)\}$. It is easy to see that $\Pi^{s_1}_{s_2}$ is not empty and that $\text{paths}(s_2) = \{\pi^n \mid \pi \in \Pi^{s_1}_{s_2}\}$.

Induction start:

- ▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$ such that $\bar{\gamma} \in \text{CTLQ}^6 \cup \text{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \text{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$ such that $\bar{\gamma} \in \text{CTLQ}^1 \cup \text{CTLQ}^2 \cup \text{CTLQ}^6 \cup \text{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \text{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we are done because θ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by

assumption resp. Lemma 4.23 and Lemma 2.1, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{U}\, \theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathring{\mathbf{U}}\, \theta) = \mathbf{A}(\bar{\gamma} \,\mathbf{U}\, (\bar{\gamma} \wedge \theta))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$. If $k < n$, we have $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l]} \models \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^k \models \bar{\gamma}[\bot]$ or $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^k \models \bar{\gamma}[\bot]$, we are done because we have $\bar{\gamma}[\bot] \wedge \theta$ which trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, we have $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+l} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathring{\mathbf{U}}\, \theta$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by Lemma 4.23 and Lemma 2.1, we obtain $\pi^{[n,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathring{\mathbf{U}}\, \theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta \,\mathbf{U}\, \bar{\gamma})$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta \,\mathbf{U}\, \bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta \,\mathring{\mathbf{U}}\, \bar{\gamma}) = \mathbf{A}(\theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta \wedge \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \theta \wedge \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because we have $\theta \wedge \bar{\gamma}[\bot]$ which trivially implies $\gamma[\bot]$,

that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta \,\mathring{\mathbf{U}}\, \bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta \,\bar{\mathbf{U}}\, \bar{\gamma}) = \mathbf{A}(\theta \,\mathbf{U}\, (\neg\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because we have $\neg\theta \wedge \bar{\gamma}[\bot]$ which trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that $\pi^{[n,n+l)} \models \theta$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta \,\bar{\mathbf{U}}\, \bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathbf{W}\, \theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, \theta))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^1 \cup \mathrm{CTLQ}^2 \cup \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$, we are done if $k < n$ because θ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption resp. Lemma 4.23 and Lemma 2.1, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{U}\, \theta$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{W}\, \theta$. (ii) Otherwise, if no such k exists, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption resp. Lemma 4.23 and Lemma 2.1, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi] \,\mathbf{W}\, \theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathring{\mathbf{W}}\, \theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathring{\mathbf{U}}\, \theta))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$

for some $n \in \mathbb{N}$. Now, let us distinguish between three cases: (i) If there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta$ and $\pi^{n+l} \models \theta$, then it holds that $\pi^{[0,k]} \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l]} \models \bar{\gamma}[\psi]$. In particular, if $k < n$, we know that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+l]} \models \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^k \models \bar{\gamma}[\bot]$ or $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^k \models \bar{\gamma}[\bot]$, we are done because we have $\bar{\gamma}[\bot] \wedge \theta$ which trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, we have $\pi^{[n,n+l]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+l} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\, \mathring{\mathbf{U}}\, \theta$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]\, \mathring{\mathbf{W}}\, \theta$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k]} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by Lemma 4.23 and Lemma 2.1, we obtain $\pi^{[n,k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\, \mathring{\mathbf{U}}\, \theta$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]\, \mathring{\mathbf{W}}\, \theta$. (ii) Otherwise, if no such l exists but there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$, then it holds that $k < n$, $\pi^{[0,k]} \models \bar{\gamma}[\varphi]$, and $\pi^n \models \mathbf{G}\, \bar{\gamma}[\psi]$. In particular, we know that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{[n,\infty)} \models \bar{\gamma}[\psi]$. Hence, by Lemma 4.23, we obtain $\pi^k \models \bar{\gamma}[\bot]$ or $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^k \models \bar{\gamma}[\bot]$, we are done because we have $\bar{\gamma}[\bot] \wedge \theta$ which trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, we have $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \gamma[\varphi \wedge \psi]\, \mathring{\mathbf{W}}\, \theta$. (iii) Otherwise, if no such k and l exist, we know that $\pi \models \mathbf{G}\, \bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\, \bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by Lemma 4.23 and Lemma 2.1, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\, \mathring{\mathbf{W}}\, \theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\, \mathbf{W}\, \bar{\gamma}) = \mathbf{A}((\mathbf{G}\, \theta) \vee (\theta\, \mathbf{U}\, \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$, we know by Lemma 4.23 that $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\, \mathbf{U}\, \bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \theta\, \mathbf{W}\, \bar{\gamma}[\varphi \wedge \psi]$.

(ii) Otherwise, if no such k and l exist, we know that $\pi^n \models \mathbf{G}\,\theta$, which trivially implies $\pi^n \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\mathring{\mathbf{U}}\,\bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \theta \wedge \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \theta \wedge \bar{\gamma}[\psi]$, we know by Lemma 4.23 that $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because we have $\theta \wedge \bar{\gamma}[\bot]$, which trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that there exists $r \in \{k, n+l\}$ such that $\pi^{[n,r]} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\,\mathring{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \theta\,\mathring{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, if no such k and l exist, we know that $\pi^n \models \mathbf{G}\,\theta$, which trivially implies $\pi^n \models \theta\,\mathring{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\bar{\mathbf{U}}\,\bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \neg\theta \wedge \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\psi]$, we know by Lemma 4.23 that $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. If $\pi^{\min(k,n+l)} \models \bar{\gamma}[\bot]$ and $k < n$, we are done because we have $\neg\theta \wedge \bar{\gamma}[\bot]$, which trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we know that $\pi^{[n,n+l)} \models \theta$ and $\pi^{n+l} \models \neg\theta \wedge \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\,\bar{\mathbf{U}}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, if no such k and l exist, we know that $\pi^n \models \mathbf{G}\,\theta$, which trivially implies $\pi^n \models \theta\,\bar{\mathbf{W}}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

Induction step:

▷ Let $\gamma = \theta \vee \bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. If $s_1 \models \theta$ or $s_2 \models \theta$, we trivially obtain $s_1 \models \gamma[\bot]$ resp. $s_2 \models \gamma[\varphi \wedge \psi]$.

Otherwise, we know that $s_1 \models \bar{\gamma}[\varphi]$ and $s_2 \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $r < n$ and $\rho^r \models \bar{\gamma}[\bot]$, which trivially imply $s_2 \models \gamma[\varphi \wedge \psi]$ and $\rho^r \models \gamma[\bot]$ respectively. Note that in the latter case it holds that $\pi^r \models \gamma[\bot]$ for all $\pi \in \Pi_{s_2}^{s_1}$.

▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi_{s_2}^{s_1}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, n+l) \leq r < \max(k, n+l)$ and $\pi^r \models \bar{\gamma}[\bot]$. If $r < n$, we are done because $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, that is, $\rho^r \models \gamma[\bot]$. Note that in this case it holds that $\pi^r \models \gamma[\bot]$ for all $\pi \in \Pi_{s_2}^{s_1}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we have $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi_{s_2}^{s_1}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi_{s_2}^{s_1}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$. If $k < n$, we are done because θ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi_{s_2}^{s_1}$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{U}\,\theta$. Thus, since $\pi \in \Pi_{s_2}^{s_1}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma})$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi_{s_2}^{s_1}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, n+l) \leq r < \max(k, n+l)$ and $\pi^r \models \bar{\gamma}[\bot]$. If $r < n$, we are done because $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, that is, $\rho^r \models \gamma[\bot]$. Note that in this case it holds that $\pi^r \models \gamma[\bot]$ for all $\pi \in \Pi_{s_2}^{s_1}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we have either $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[n,\max(k,n+l))} \models \theta$ and $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi_{s_2}^{s_1}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{W}\,\theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathbf{U}\,\theta))$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exists a least $k \in \mathbb{N}$ such that $\pi^k \models \theta$, we are done if $k < n$ because θ trivially implies $\gamma[\bot]$, that is, $\rho^k \models \gamma[\bot]$. Note that in this case it holds that $\pi^k \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, if $k \geq n$, we know that $\pi^{[n,k)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,k)} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^k \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{U}\,\theta$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{W}\,\theta$. (ii) Otherwise, if no such k exists, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathbf{W}\,\theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,\bar{\gamma}))$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$, we obtain by induction hypothesis $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$ or an $r \in \mathbb{N}$ such that $\min(k, n+l) \leq r < \max(k, n+l)$ and $\pi^r \models \bar{\gamma}[\bot]$. If $r < n$, we are done because $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, that is, $\rho^r \models \gamma[\bot]$. Note that in this case it holds that $\pi^r \models \gamma[\bot]$ for all $\pi \in \Pi^{s_1}_{s_2}$. Otherwise, since $\bar{\gamma}[\bot]$ implies $\bar{\gamma}[\varphi \wedge \psi]$ by Lemma 2.1, we have either $\pi^{[n,r)} \models \theta$ and $\pi^r \models \bar{\gamma}[\varphi \wedge \psi]$ or $\pi^{[n,\max(k,n+l))} \models \theta$ and $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, if no such k and l exist, we know that $\pi^n \models \mathbf{G}\,\theta$, which trivially implies $\pi^n \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we obtain $s_2 \models \gamma[\varphi \wedge \psi]$.

This concludes the proof. □

Proof of Lemma 4.26

Lemma 4.26. *Let $\gamma \in CTLQ^8 \cup CTLQ^9$. Suppose that every subquery in $CTLQ^1$, $CTLQ^3$, $CTLQ^4$, $CTLQ^6$, and $CTLQ^7$ is weak collecting. Then, γ is strong collecting.*

Proof. Structural induction on γ.
Let s_1 and s_2 be any states in a Kripke structure such that $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\rho \in \text{paths}(s_1)$ such that $\rho(n) = s_2$ for some $n \in \mathbb{N}$. Let us define the set $\Pi^{s_1}_{s_2} = \{\pi \in \text{paths}(s_1) \mid \forall i \leq n.\ \pi(i) = \rho(i)\}$. It is easy to see that $\Pi^{s_1}_{s_2}$ is not empty and that $\text{paths}(s_2) = \{\pi^n \mid \pi \in \Pi^{s_1}_{s_2}\}$.

Induction start:

- ▷ Let $\gamma = \mathbf{AG}\,\bar{\gamma}$ such that $\bar{\gamma} \in \text{CTLQ}^1 \cup \text{CTLQ}^3 \cup \text{CTLQ}^4 \cup \text{CTLQ}^6 \cup \text{CTLQ}^7$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \text{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by assumption, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

Induction step:

- ▷ Let $\gamma = \theta \wedge \bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. Then, we know that $s_1 \models \bar{\gamma}[\varphi]$ and $s_2 \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$. Thus, we have $s_2 \models \theta$ and $s_2 \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $s_2 \models \gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \text{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that $\pi^1 \models \bar{\gamma}[\varphi]$ and $\pi^{n+1} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{n+1} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{X}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \text{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{F}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

- ▷ Let $\gamma = \mathbf{AG}\,\bar{\gamma}$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have

$\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that $\pi \models \mathbf{G}\,\bar{\gamma}[\varphi]$ and $\pi^n \models \mathbf{G}\,\bar{\gamma}[\psi]$, which implies $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi] \wedge \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta) = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta))$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that $\pi \models \bar{\gamma}[\varphi]$ and that there exists a least $k \in \mathbb{N}$ such that $\pi^{n+k} \models \theta$ and therefore $\pi^{[n,n+k]} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,n+k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,n+k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+k} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma})$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Then, we know that there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,\max(k,n+l))} \models \theta$ and $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{U}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{U}\,(\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the strong until operator $\mathbf{U}$ with respect to its second argument, we are done.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{U}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{U}\,(\neg\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the strong until operator $\mathbf{U}$ with respect to its second argument, we are done.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{W}}\,\theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta))$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exists a least $k \in \mathbb{N}$ such that $\pi^{n+k} \models \theta$, we know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^{[n,n+k]} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,n+k]} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,n+k]} \models \bar{\gamma}[\varphi \wedge \psi]$ and $\pi^{n+k} \models \theta$, that is, $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{U}}\,\theta$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{W}}\,\theta$. (ii) Otherwise, if no such k exists, we

know that $\pi \models \bar{\gamma}[\varphi]$ and $\pi^{[n,\infty)} \models \bar{\gamma}[\psi]$. Hence, by induction hypothesis, we obtain $\pi^{[n,\infty)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \mathbf{G}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \bar{\gamma}[\varphi \wedge \psi]\,\mathring{\mathbf{W}}\,\theta$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{W}\,\bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta\,\mathbf{U}\,\bar{\gamma}))$. Suppose that $s_1 \models \gamma[\varphi]$ and $s_2 \models \gamma[\psi]$, where $s_1 \rightsquigarrow s_2$. W.l.o.g., we choose any path $\pi \in \Pi^{s_1}_{s_2}$. Recall that by definition we have $\pi^n \in \mathrm{paths}(s_2)$ for some $n \in \mathbb{N}$. Now, let us distinguish between two cases: (i) If there exist least $k, l \in \mathbb{N}$ such that $\pi^k \models \bar{\gamma}[\varphi]$ and $\pi^{n+l} \models \bar{\gamma}[\psi]$, we obtain by induction hypothesis $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$. So we have $\pi^{[n,\max(k,n+l))} \models \theta$ and $\pi^{\max(k,n+l)} \models \bar{\gamma}[\varphi \wedge \psi]$, that is, $\pi^n \models \theta\,\mathbf{U}\,\bar{\gamma}[\varphi \wedge \psi]$, which trivially implies $\pi^n \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi \wedge \psi]$. (ii) Otherwise, if no such k and l exist, we know that $\pi^n \models \mathbf{G}\,\theta$, which trivially implies $\pi^n \models \theta\,\mathbf{W}\,\bar{\gamma}[\varphi \wedge \psi]$. Thus, since $\pi \in \Pi^{s_1}_{s_2}$ was chosen w.l.o.g., we have $s_2 \models \gamma[\varphi \wedge \psi]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{W}\,(\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the weak until operator $\mathbf{W}$ with respect to its second argument, we are done.

▷ Let $\gamma = \mathbf{A}(\theta\,\bar{\mathbf{W}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{W}\,(\neg\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the weak until operator $\mathbf{W}$ with respect to its second argument, we are done.

This concludes the proof. □

Proof of Lemma 4.31

Lemma 4.31. *Let $\gamma \in CTLQ^3 \cup CTLQ^6$. Then, $\gamma[\top]$ implies $\mathbf{AF}\,\gamma[\bot]$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$ such that $\bar{\gamma} \in \mathrm{CTLQ}^1 \cup \mathrm{CTLQ}^2 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^5 \cup \mathrm{CTLQ}^7 \cup \mathrm{CTLQ}^8 \cup \mathrm{CTLQ}^9$. If $s \models \gamma[\top]$, we choose w.l.o.g. any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists $n \in \mathbb{N}$ such that $\pi^n \models \theta$. Hence, since θ trivially implies $\gamma[\bot]$, we obtain $\pi^n \models \gamma[\bot]$, that is, $\pi \models \mathbf{F}\,\gamma[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \mathbf{AF}\,\gamma[\bot]$.

Induction step:

▷ Let $\gamma = \theta \vee \bar{\gamma}$. If $s \models \gamma[\top]$ and $s \models \theta$, we trivially have $s \models \gamma[\bot]$ and therefore $s \models \mathbf{AF}\,\gamma[\bot]$. Otherwise, if $s \models \gamma[\top]$ and $s \not\models \theta$, we know that $s \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $s \models \mathbf{AF}\,\bar{\gamma}[\bot]$. Thus, since $\bar{\gamma}[\bot]$ trivially implies $\gamma[\bot]$, we have $s \models \mathbf{AF}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$. If $s \models \gamma[\top]$, we choose w.l.o.g. any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists $n \in \mathbb{N}$ such that $\pi^n \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^n \models \mathbf{AF}\,\bar{\gamma}[\bot]$, which implies $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$. Consequently, since $\bar{\gamma}[\bot]$ implies $\gamma[\bot]$, we have $\pi \models \mathbf{F}\,\gamma[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \mathbf{AF}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$. If $s \models \gamma[\top]$, we choose w.l.o.g. any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists $n \in \mathbb{N}$ such that $\pi^n \models \theta$. Hence, since θ trivially implies $\gamma[\bot]$, we obtain $\pi^n \models \gamma[\bot]$, that is, $\pi \models \mathbf{F}\,\gamma[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \mathbf{AF}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma})$. If $s \models \gamma[\top]$, we choose w.l.o.g. any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists $n \in \mathbb{N}$ such that $\pi^n \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^n \models \mathbf{AF}\,\bar{\gamma}[\bot]$, which implies $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$. Consequently, since $\bar{\gamma}[\bot]$ implies $\gamma[\bot]$, we have $\pi \models \mathbf{F}\,\gamma[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \mathbf{AF}\,\gamma[\bot]$.

This concludes the proof. □

Proof of Lemma 4.32

Lemma 4.32. *Let* $\gamma \in CTLQ^2 \cup CTLQ^5 \cup CTLQ^9$. *Then,* $\mathbf{AG}\,\gamma[\top]$ *implies* $\mathbf{AG}\,\gamma[\bot]$.

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \mathbf{A}(\theta\,\mathring{\mathbf{U}}\,\bar{\gamma}) = \mathbf{A}(\theta\,\mathbf{U}\,(\theta \wedge \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^6$. It is easy to see that $\mathbf{AG}\,\gamma[\top]$ implies both $\mathbf{AG}\,\theta$ and $\mathbf{AG}\,\mathbf{AF}\,\bar{\gamma}[\top]$. Hence, by Lemma 4.31, we obtain $\mathbf{AG}\,\mathbf{AF}\,\mathbf{AF}\,\bar{\gamma}[\bot]$, which is equivalent to $\mathbf{AG}\,\mathbf{AF}\,\bar{\gamma}[\bot]$. So we have $\mathbf{AG}\,\theta$ and $\mathbf{AG}\,\mathbf{AF}\,\bar{\gamma}[\bot]$, that is, $\mathbf{AG}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta \mathring{\mathbf{W}} \bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta \mathring{\mathbf{U}} \bar{\gamma}))$ such that $\bar{\gamma} \in \mathrm{CTLQ}^3 \cup \mathrm{CTLQ}^4 \cup \mathrm{CTLQ}^6 \cup \mathrm{CTLQ}^7 \cup \mathrm{CTLQ}^8$. It is easy to see that $\mathbf{AG}\,\gamma[\top]$ implies $\mathbf{AG}\,\theta$, which in turn trivially implies $\mathbf{AG}\,\gamma[\bot]$.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. Since $\mathbf{AG}\,\gamma[\top]$ is equivalent to $(\mathbf{AG}\,\theta) \wedge (\mathbf{AG}\,\bar{\gamma}[\top])$, we obtain by induction hypothesis $(\mathbf{AG}\,\theta) \wedge (\mathbf{AG}\,\bar{\gamma}[\bot])$. Hence, we have $\mathbf{AG}\,(\theta \wedge \bar{\gamma}[\bot])$, that is, $\mathbf{AG}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$. Since $\mathbf{AG}\,\gamma[\top]$ is equivalent to $\mathbf{AX}\,\mathbf{AG}\,\bar{\gamma}[\top]$, we obtain by induction hypothesis $\mathbf{AX}\,\mathbf{AG}\,\bar{\gamma}[\bot]$. Hence, we have $\mathbf{AG}\,\mathbf{AX}\,\bar{\gamma}[\bot]$, that is, $\mathbf{AG}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \mathring{\mathbf{U}} \theta) = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta))$. It is easy to see that $\mathbf{AG}\,\gamma[\top]$ implies both $\mathbf{AG}\,\bar{\gamma}[\top]$ and $\mathbf{AG}\,\mathbf{AF}\,\theta$. Hence, by induction hypothesis, we obtain $\mathbf{AG}\,\bar{\gamma}[\bot]$. So we have $\mathbf{AG}\,\bar{\gamma}[\bot]$ and $\mathbf{AG}\,\mathbf{AF}\,\theta$, that is, $\mathbf{AG}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \mathring{\mathbf{W}} \theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \mathring{\mathbf{U}} \theta))$. It is easy to see that $\mathbf{AG}\,\gamma[\top]$ implies $\mathbf{AG}\,\bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\mathbf{AG}\,\bar{\gamma}[\bot]$. Thus, since $\mathbf{AG}\,\bar{\gamma}[\bot]$ is equivalent to $\mathbf{AG}\,\mathbf{AG}\,\bar{\gamma}[\bot]$, and $\mathbf{AG}\,\bar{\gamma}[\bot]$ implies $\gamma[\bot]$, we have $\mathbf{AG}\,\gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta \mathring{\mathbf{W}} \bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta \mathring{\mathbf{U}} \bar{\gamma}))$. It is easy to see that $\mathbf{AG}\,\gamma[\top]$ implies $\mathbf{AG}\,\theta$. Hence, since $\mathbf{AG}\,\theta$ trivially implies $\mathbf{AG}\,\gamma[\bot]$, we obtain $\mathbf{AG}\,\gamma[\bot]$.

This concludes the proof. □

Proof of Lemma 4.33

Lemma 4.33. *Let $\gamma \in CTLQ^{10}$. Then, $\gamma[\top]$ implies $\gamma[\bot]$.*

Proof. Structural induction on γ.

Induction start:

▷ Let $\gamma = \mathbf{AG}\,\bar{\gamma}$ such that $\bar{\gamma} \in \mathrm{CTLQ}^2 \cup \mathrm{CTLQ}^5 \cup \mathrm{CTLQ}^9$. Then, the assertion holds according to Lemma 4.32.

Induction step:

▷ Let $\gamma = \theta \wedge \bar{\gamma}$. Since $\gamma[\top]$ is equivalent to $\theta \wedge \bar{\gamma}[\top]$, we obtain by induction hypothesis $\theta \wedge \bar{\gamma}[\bot]$, that is, $\gamma[\bot]$.

▷ Let $\gamma = \theta \vee \bar{\gamma}$. Since $\gamma[\top]$ is equivalent to $\theta \vee \bar{\gamma}[\top]$, we obtain by induction hypothesis $\theta \vee \bar{\gamma}[\bot]$, that is, $\gamma[\bot]$.

▷ Let $\gamma = \mathbf{AX}\,\bar{\gamma}$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that $\pi^1 \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^1 \models \bar{\gamma}[\bot]$, that is, $\pi \models \mathbf{X}\,\bar{\gamma}[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{AF}\,\bar{\gamma}$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists $n \in \mathbb{N}$ such that $\pi^n \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^n \models \bar{\gamma}[\bot]$, that is, $\pi \models \mathbf{F}\,\bar{\gamma}[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{AG}\,\bar{\gamma}$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,\infty)} \models \bar{\gamma}[\bot]$, that is, $\pi \models \mathbf{G}\,\bar{\gamma}[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,\theta)$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \theta$ and therefore $\pi^{[0,n)} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,n)} \models \bar{\gamma}[\bot]$. So we have $\pi^{[0,n)} \models \bar{\gamma}[\bot]$ and $\pi^n \models \theta$, that is, $\pi \models \bar{\gamma}[\bot]\,\mathbf{U}\,\theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma}\,\mathring{\mathbf{U}}\,\theta) = \mathbf{A}(\bar{\gamma}\,\mathbf{U}\,(\bar{\gamma} \wedge \theta))$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \theta$ and therefore $\pi^{[0,n]} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,n]} \models \bar{\gamma}[\bot]$. So we have $\pi^{[0,n]} \models \bar{\gamma}[\bot]$ and $\pi^n \models \theta$, that is, $\pi \models \bar{\gamma}[\bot]\,\mathring{\mathbf{U}}\,\theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta\,\mathbf{U}\,\bar{\gamma})$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Then, we know that there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^n \models \bar{\gamma}[\bot]$. So

we have $\pi^{[0,n)} \models \theta$ and $\pi^n \models \bar{\gamma}[\bot]$, that is, $\pi \models \theta \,\mathbf{U}\, \bar{\gamma}[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta \,\mathring{\mathbf{U}}\, \bar{\gamma}) = \mathbf{A}(\theta \,\mathbf{U}\, (\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the strong until operator $\mathbf{U}$ with respect to its second argument, we are done.

▷ Let $\gamma = \mathbf{A}(\theta \,\bar{\mathbf{U}}\, \bar{\gamma}) = \mathbf{A}(\theta \,\mathbf{U}\, (\neg\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the strong until operator $\mathbf{U}$ with respect to its second argument, we are done.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathbf{W}\, \theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathbf{U}\, \theta))$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \theta$, we know that $\pi^{[0,n)} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,n)} \models \bar{\gamma}[\bot]$. So we have $\pi^{[0,n)} \models \bar{\gamma}[\bot]$ and $\pi^n \models \theta$, that is, $\pi \models \bar{\gamma}[\bot] \,\mathbf{U}\, \theta$, which trivially implies $\pi \models \bar{\gamma}[\bot] \,\mathbf{W}\, \theta$. (ii) Otherwise, if no such n exists, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,\infty)} \models \bar{\gamma}[\bot]$, that is, $\pi \models \mathbf{G}\,\bar{\gamma}[\bot]$, which trivially implies $\pi \models \bar{\gamma}[\bot] \,\mathbf{W}\, \theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\bar{\gamma} \,\mathring{\mathbf{W}}\, \theta) = \mathbf{A}((\mathbf{G}\,\bar{\gamma}) \vee (\bar{\gamma} \,\mathring{\mathbf{U}}\, \theta))$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \theta$, we know that $\pi^{[0,n]} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,n]} \models \bar{\gamma}[\bot]$. So we have $\pi^{[0,n]} \models \bar{\gamma}[\bot]$ and $\pi^n \models \theta$, that is, $\pi \models \bar{\gamma}[\bot] \,\mathring{\mathbf{U}}\, \theta$, which trivially implies $\pi \models \bar{\gamma}[\bot] \,\mathring{\mathbf{W}}\, \theta$. (ii) Otherwise, if no such n exists, we know that $\pi^{[0,\infty)} \models \bar{\gamma}[\top]$. Hence, by induction hypothesis, we obtain $\pi^{[0,\infty)} \models \bar{\gamma}[\bot]$, that is, $\pi \models \mathbf{G}\,\bar{\gamma}[\bot]$, which trivially implies $\pi \models \bar{\gamma}[\bot] \,\mathring{\mathbf{W}}\, \theta$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta \,\mathbf{W}\, \bar{\gamma}) = \mathbf{A}((\mathbf{G}\,\theta) \vee (\theta \,\mathbf{U}\, \bar{\gamma}))$. Suppose that $s \models \gamma[\top]$. W.l.o.g., we choose any path $\pi \in \mathrm{paths}(s)$. Now, let us distinguish between two cases: (i) If there exists a least $n \in \mathbb{N}$ such that $\pi^n \models \bar{\gamma}[\top]$, we obtain by induction hypothesis $\pi^n \models \bar{\gamma}[\bot]$. So we have $\pi^{[0,n)} \models \theta$ and $\pi^n \models \bar{\gamma}[\bot]$, that is, $\pi \models \theta \,\mathbf{U}\, \bar{\gamma}[\bot]$, which trivially implies $\pi \models \theta \,\mathbf{W}\, \bar{\gamma}[\bot]$. (ii) Otherwise, if no such n exists, we know that $\pi \models \mathbf{G}\,\theta$, which trivially implies $\pi \models \theta \,\mathbf{W}\, \bar{\gamma}[\bot]$. Thus, since $\pi \in \mathrm{paths}(s)$ was chosen w.l.o.g., we have $s \models \gamma[\bot]$.

▷ Let $\gamma = \mathbf{A}(\theta \,\mathring{\mathbf{W}}\, \bar{\gamma}) = \mathbf{A}(\theta \,\mathbf{W}\, (\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the weak until operator $\mathbf{W}$ with respect to its second argument, we are done.

▷ Let $\gamma = \mathbf{A}(\theta \,\bar{\mathbf{W}}\, \bar{\gamma}) = \mathbf{A}(\theta \,\mathbf{W}\, (\neg\theta \wedge \bar{\gamma}))$. Since we have already shown the assertion for conjunction and the weak until operator $\mathbf{W}$ with respect to its second argument, we are done.

This concludes the proof. □

Bibliography

[AENT03] Nina Amla, E. Allen Emerson, Kedar S. Namjoshi, and Richard J. Trefler. Abstract patterns of compositional reasoning. In *Proceedings of the 14th International Conference on Concurrency Theory (CONCUR)*, volume 2761 of *Lecture Notes in Computer Science*, pages 423–438. Springer-Verlag, 2003.

[AFF+03] Roy Armoni, Limor Fix, Alon Flaisher, Orna Grumberg, Nir Piterman, Andreas Tiemeyer, and Moshe Y. Vardi. Enhanced vacuity detection in linear temporal logic. In *Proceedings of the 15th International Conference on Computer Aided Verification (CAV)*, volume 2725 of *Lecture Notes in Computer Science*, pages 368–380. Springer-Verlag, 2003.

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[AI03] Micah Adler and Neil Immerman. An $n!$ lower bound on formula size. *ACM Transactions on Computational Logic (TOCL)*, 4(3):296–314, 2003.

[AL93] Martín Abadi and Leslie Lamport. Composing specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 15(1):73–132, 1993.

[AL95] Martín Abadi and Leslie Lamport. Conjoining specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 17(3):507–534, 1995.

[BB94] Derek L. Beatty and Randal E. Bryant. Formally verifying a microprocessor using a simulation methodology. In *Proceedings of the 31st Conference on Design Automation (DAC)*, pages 596–602. ACM, 1994.

[BBDER97] Ilan Beer, Shoham Ben-David, Cindy Eisner, and Yoav Rodeh. Efficient detection of vacuity in ACTL formulas. In *Proceedings of the 9th International Conference on Computer Aided Verification (CAV)*, volume 1254 of *Lecture Notes in Computer Science*, pages 279–290. Springer-Verlag, 1997.

[BBDER01] Ilan Beer, Shoham Ben-David, Cindy Eisner, and Yoav Rodeh. Efficient detection of vacuity in temporal model checking. *Formal Methods in System Design*, 18(2):141–163, 2001.

[BCCZ99] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu. Symbolic model checking without BDDs. In *Proceedings of the 5th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 1579 of *Lecture Notes in Computer Science*, pages 193–207. Springer-Verlag, 1999.

[BCL+94] Jerry R. Burch, Edmund M. Clarke, David E. Long, Kenneth L. McMillan, and David L. Dill. Symbolic model checking for sequential circuit verification. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13(4):401–424, 1994.

[BCM+92] Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170, 1992.

[BdRV01] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. Number 53 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[BEGL01] Francesco Buccafurri, Thomas Eiter, Georg Gottlob, and Nicola Leone. On ACTL formulas having linear counterexam-

ples. *Journal of Computer and System Sciences*, 62(3):463–515, 2001.

[BFG+] Doron Bustan, Alon Flaisher, Orna Grumberg, Orna Kupferman, and Moshe Y. Vardi. Regular vacuity. Submitted for publication.

[BG01] Glenn Bruns and Patrice Godefroid. Temporal logic query checking. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 409–417. IEEE Computer Society, 2001.

[Bry86] Randal E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986.

[Bry92] Randal E. Bryant. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, 1992.

[BS84] Robert A. Bull and Krister Segerberg. Basic modal logic. In Dov M. Gabbay and Franz Guenthner, editors, *Handbook of Philosophical Logic*, volume II: Extensions of Classical Logic, chapter 1, pages 1–88. D. Reidel Publishing Company, 1984.

[CD88] Edmund M. Clarke and Ioana A. Draghicescu. Expressibility results for linear time and branching time logics. In *Proceedings of the REX Workshop on Linear Time, Branching Time, and Partial Order in Logics and Models for Concurrency*, volume 354 of *Lecture Notes in Computer Science*, pages 428–437. Springer-Verlag, 1988.

[CE82] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Proceedings of the Workshop on Logic of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer-Verlag, 1982.

[CES86] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 8(2):244–263, 1986.

[CG03] Marsha Chechik and Arie Gurfinkel. TLQSolver: A temporal logic query checker. In *Proceedings of the 15th International Conference on Computer Aided Verification (CAV)*, volume 2725 of *Lecture Notes in Computer Science*, pages 210–214. Springer-Verlag, 2003.

[CGH+95] Edmund M. Clarke, Orna Grumberg, Hiromi Hiraishi, Somesh Jha, David E. Long, Kenneth L. McMillan, and Linda A. Ness. Verification of the Futurebus+ cache coherence protocol. *Formal Methods in System Design*, 6(2):217–232, 1995.

[CGJ+03] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM*, 50(5):752–794, 2003.

[CGL94] Edmund M. Clarke, Orna Grumberg, and David E. Long. Model checking and abstraction. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 16(5):1512–1542, 1994.

[CGP99] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 1999.

[Cha00] William Chan. Temporal-logic queries. In *Proceedings of the 12th International Conference on Computer Aided Verification (CAV)*, volume 1855 of *Lecture Notes in Computer Science*, pages 450–463. Springer-Verlag, 2000.

[CJLV02] Edmund M. Clarke, Somesh Jha, Yuan Lu, and Helmut Veith. Tree-like counterexamples in model checking. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 19–29. IEEE Computer Society, 2002.

[CLM89] Edmund M. Clarke, David E. Long, and Kenneth L. McMillan. Compositional model checking. In *Proceedings of the 4th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 353–362. IEEE Computer Society, 1989.

[CS00] Edmund M. Clarke and Pasupathi A. Subrahmanyam. Hardware verification. In Anthony Ralston, Edwin D. Reilly, and David Hemmendinger, editors, *Encyclopedia of Computer Science*, pages 777–780. Nature Publishing Group, fourth edition, 2000.

[CS01] Edmund M. Clarke and Bernd-Holger Schlingloff. Model checking. In J. Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 24, pages 1635–1790. Elsevier Science, 2001.

[dRdBH+01] Willem-Paul de Roever, Frank de Boer, Ulrich Hanneman, Jozef Hooman, Yassine Lakhnech, Mannes Poel, and Job Zwiers. *Concurrency Verification: Introduction to Compositional and Noncompositional Methods.* Number 54 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[EC80] E. Allen Emerson and Edmund M. Clarke. Characterizing correctness properties of parallel programs using fixpoints. In *Proceedings of the 7th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 85 of *Lecture Notes in Computer Science*, pages 169–181. Springer-Verlag, 1980.

[EC82] E. Allen Emerson and Edmund M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Science of Computer Programming*, 2(3):241–266, 1982.

[EH85] E. Allen Emerson and Joseph Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30(1):1–24, 1985.

[EH86] E. Allen Emerson and Joseph Y. Halpern. "Sometimes" and "Not Never" revisited: On branching versus linear time temporal logic. *Journal of the ACM*, 33(1):151–178, 1986.

[EJ88] E. Allen Emerson and Charanjit S. Jutla. The complexity of tree automata and logics of programs. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 328–337. IEEE Computer Society, 1988.

[Eme90] E. Allen Emerson. Temporal and modal logic. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Methods and Semantics, chapter 16, pages 995–1067. Elsevier Science, 1990.

[Gab89] Dov M. Gabbay. The declarative past and imperative future: Executable temporal logic for interactive systems. In *Temporal Logic in Specification*, volume 398 of *Lecture Notes in Computer Science*, pages 409–448. Springer-Verlag, 1989.

[GC04a] Arie Gurfinkel and Marsha Chechik. Extending extended vacuity. In *Proceedings of the 5th International Conference on Formal Methods in Computer-Aided Design (FMCAD)*, Lecture Notes in Computer Science. Springer-Verlag, 2004. To appear.

[GC04b] Arie Gurfinkel and Marsha Chechik. How vacuous is vacuous? In *Proceedings of the 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 2988 of *Lecture Notes in Computer Science*, pages 451–466. Springer-Verlag, 2004.

[GCD03] Arie Gurfinkel, Marsha Chechik, and Benet Devereux. Temporal logic query checking: A tool for model exploration. *IEEE Transactions on Software Engineering (TSE)*, 29(10):898–914, 2003.

[GDC02] Arie Gurfinkel, Benet Devereux, and Marsha Chechik. Model exploration with temporal logic query checking. In *Proceedings of the 10th ACM Symposium on Foundations of Software Engineering (FSE)*, pages 139–148. ACM, 2002.

[GHR94] Dov M. Gabbay, Ian Hodkinson, and Mark A. Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects*, volume 1. Number 28 in Oxford Logic Guides. Oxford University Press, 1994.

[GL94] Orna Grumberg and David E. Long. Model checking and modular verification. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 16(3):843–871, 1994.

[GRF00] Dov M. Gabbay, Mark A. Reynolds, and Marcelo Finger. *Temporal Logic: Mathematical Foundations and Computational Aspects*, volume 2. Number 40 in Oxford Logic Guides. Oxford University Press, 2000.

[GT96] Winfried K. Grassmann and Jean-Paul Tremblay. *Logic and Discrete Mathematics.* Prentice Hall, 1996.

[HC98] George E. Hughes and Max J. Cresswell. *A New Introduction to Modal Logic.* Routledge, 1998.

[HMU01] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley, second edition, 2001.

[Hoa69] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580,583, 1969.

[HS02] Samuel Hornus and Philippe Schnoebelen. On solving temporal logic queries. In *Proceedings of the 9th International Conference on Algebraic Methodology and Software Technology (AMAST)*, volume 2422 of *Lecture Notes in Computer Science*, pages 163–177. Springer-Verlag, 2002.

[JL03] Jan Johannsen and Martin Lange. CTL^+ is complete for double exponential time. In *Proceedings of the 30th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2719 of *Lecture Notes in Computer Science*, pages 767–775. Springer-Verlag, 2003.

[Kam68] Johan A. W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, 1968.

[Koz83] Dexter Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[Kri63a] Saul A. Kripke. Semantical analysis of modal logic I: Normal propositional calculi. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

[Kri63b] Saul A. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[KV99] Orna Kupferman and Moshe Y. Vardi. Vacuity detection in temporal model checking. In *Proceedings of the 10th Advanced Research Working Conference on Correct Hardware Design and Verification Methods (CHARME)*, volume 1703 of *Lecture Notes in Computer Science*, pages 82–96. Springer-Verlag, 1999.

[KVW00] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000.

[Lam80] Leslie Lamport. "Sometime" is sometimes "not never" – On the temporal logic of programs. In *Proceedings of the 7th Annual ACM Symposium on Principles of Programming Languages (POPL)*, pages 174–185. ACM, 1980.

[LC00] Ralph L. London and Daniel Craigen. Program verification. In Anthony Ralston, Edwin D. Reilly, and David Hemmendinger, editors, *Encyclopedia of Computer Science*, pages 1458–1461. Nature Publishing Group, fourth edition, 2000.

[LMS01] François Laroussinie, Nicolas Markey, and Philippe Schnoebelen. Model checking CTL^+ and FCTL is hard. In *Proceedings of the 4th International Conference on Foundations of Software Science and Computation Structures (FOSSACS)*, volume 2030 of *Lecture Notes in Computer Science*, pages 318–331. Springer-Verlag, 2001.

[LMS02] François Laroussinie, Nicolas Markey, and Philippe Schnoebelen. Temporal logic with forgettable past. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 383–392. IEEE Computer Society, 2002.

[LP85] Orna Lichtenstein and Amir Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *Proceedings of the 12th Annual ACM Symposium on Principles of Programming Languages (POPL)*, pages 97–107. ACM, 1985.

[Mai00] Monika Maidl. The common fragment of CTL and LTL. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 643–652. IEEE Computer Society, 2000.

[Mai01] Patrick Maier. A set-theoretic framework for assume-guarantee reasoning. In *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of *Lecture Notes in Computer Science*, pages 821–834. Springer-Verlag, 2001.

[MC81] Jayadev Misra and K. Mani Chandy. Proofs of networks of processes. *IEEE Transactions on Software Engineering (TSE)*, 7(4):417–426, 1981.

[McM93] Kenneth L. McMillan. *Symbolic Model Checking.* Kluwer Academic Publishers, 1993.

[McM99] Kenneth L. McMillan. Circular compositional reasoning about liveness. In *Proceedings of the 10th Advanced Research Working Conference on Correct Hardware Design and Verification Methods (CHARME)*, volume 1703 of *Lecture Notes in Computer Science*, pages 342–346. Springer-Verlag, 1999.

[McM00] Kenneth L. McMillan. Model checking. In Anthony Ralston, Edwin D. Reilly, and David Hemmendinger, editors, *Encyclopedia of Computer Science*, pages 1177–1181. Nature Publishing Group, fourth edition, 2000.

[MP92] Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, 1992.

[ØH95] Peter Øhrstrøm and Per F. V. Hasle. *Temporal Logic: From Ancient Ideas to Artificial Intelligence*, volume 57 of *Studies in Linguistics and Philosophy.* Kluwer Academic Publishers, 1995.

[Pnu77] Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 46–57. IEEE Computer Society, 1977.

[Pnu81] Amir Pnueli. The temporal semantics of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.

[Pnu85] Amir Pnueli. In transition from global to modular temporal reasoning about programs. In *Logics and Models of Concurrent Systems*, volume 13 of *NATO ASI Series, Sub-Series F: Computer and System Science*, pages 123–144. Springer-Verlag, 1985.

[Pnu97] Amir Pnueli. At the 9th International Conference on Computer Aided Verification (CAV), 1997. Question from the audience (cited from [BBDER01]).

[Pri57] Arthur N. Prior. *Time and Modality.* Clarendon Press, 1957.

[QS82] Jean-Pierre Queille and Joseph Sifakis. Specification and verification of concurrent systems in CESAR. In *Proceedings of the 5th International Symposium on Programming*, volume 137 of *Lecture Notes in Computer Science*, pages 337–351. Springer-Verlag, 1982.

[Sam02] Marko Samer. Temporal logic queries in model checking. Master's thesis, Vienna University of Technology, May 2002.

[SC85] A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985.

[Sch03] Philippe Schnoebelen. The complexity of temporal logic model checking. In *Advances in Modal Logic*, volume 4, pages 393–436. King's College Publications, 2003.

[Sta85] Eugene W. Stark. A proof technique for rely/guarantee properties. In *Proceedings of the 5th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 206 of *Lecture Notes in Computer Science*, pages 369–391. Springer-Verlag, 1985.

[Sti93] Colin Stirling. Modal and temporal logics. In Samson Abramsky, Dov M. Gabbay, and Thomas S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2: Background: Computational Structures, chapter 5, pages 477–563. Oxford University Press, 1993.

[Sti01] Colin Stirling. *Modal and Temporal Properties of Processes.* Texts in Computer Science. Springer-Verlag, 2001.

[SV03] Marko Samer and Helmut Veith. Validity of CTL queries revisited. In *Proceedings of the 12th Annual Conference of the European Association for Computer Science Logic (CSL)*, volume 2803 of *Lecture Notes in Computer Science*, pages 470–483. Springer-Verlag, 2003.

[SV04a] Marko Samer and Helmut Veith. Parameterized vacuity. In *Proceedings of the 5th International Conference on Formal Methods in Computer-Aided Design (FMCAD)*, Lecture Notes in Computer Science. Springer-Verlag, 2004. To appear.

[SV04b] Marko Samer and Helmut Veith. A syntactic characterization of distributive LTL queries. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3142 of *Lecture Notes in Computer Science*, pages 1099–1110. Springer-Verlag, 2004.

[Tar55] Alfred Tarski. A lattice-theoretical theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[vB95] Johan van Benthem. Temporal logic. In Dov M. Gabbay, Christopher J. Hogger, and J. Alan Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 4: Epistemic and Temporal Reasoning, chapter 5, pages 241–350. Oxford University Press, 1995.

[VV01] Mahesh Viswanathan and Ramesh Viswanathan. Foundations for circular compositional reasoning. In *Proceedings of the 28th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of *Lecture Notes in Computer Science*, pages 835–847. Springer-Verlag, 2001.

[VW86] Moshe Y. Vardi and Pierre Wolper. An automata-theoretic approach to automatic program verification. In *Proceedings of the 1st Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 332–344. IEEE Computer Society, 1986.

[WF02] Dolores R. Wallace and Roger U. Fujii. Verification and validation. In John J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 2, pages 1815–1842. John Wiley & Sons, second edition, 2002.

[Wil99] Thomas Wilke. CTL^+ is exponentially more succinct than CTL. In *Proceedings of the 19th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 1738 of *Lecture Notes in Computer Science*, pages 110–121. Springer-Verlag, 1999.

[Win00] Jeannette M. Wing. Program specification. In Anthony Ralston, Edwin D. Reilly, and David Hemmendinger, editors, *Encyclopedia of Computer Science*, pages 1454–1458. Nature Publishing Group, fourth edition, 2000.

Index

About the Author

Marko Samer was born in November 1977 in Oberwart (Burgenland, Austria). From 1992 to 1997, he attended the Department of Electronic Data Processing and Organization at HTBL Pinkafeld (Federal College of Technology), where he graduated with honors. Afterwards, he studied Computer Science at Vienna University of Technology and Mathematics at Vienna University. In 2002, he received his Diplom-Ingenieur degree (comparable to MSc) in Computer Science with honors and started his doctoral studies under the supervision of Prof. Helmut Veith.

Marko Samer was awarded performance scholarships from both Vienna University of Technology and Vienna University. During his studies, he worked as a teaching assistant at Vienna University of Technology and as an intern in the area of compiler optimization. He acted as referee for and published several papers at international conferences.