TU WIEN Informatics

Causality Modeling and Acquisition for explainable Cyber Physical Systems

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Matthias Deimel, BSc
Matrikelnummer 01427725

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Projektass.(FWF) Reka Marta Sabou, MSc PhD
Mitwirkung: Dr.techn. Mag. Fajar Juang Ekaputra
Projektass. Mag. Peb Ruswono Aryan

Wien, 12. Juli 2021

Matthias Deimel | Reka Marta Sabou

TU WIEN Informatics

Causality Modeling and Acquisition for explainable Cyber Physical Systems

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Matthias Deimel, BSc
Registration Number 01427725

to the Faculty of Informatics

at the TU Wien

Advisor: Projektass.(FWF) Reka Marta Sabou, MSc PhD
Assistance: Dr.techn. Mag. Fajar Juang Ekaputra
Projektass. Mag. Peb Ruswono Aryan

Vienna, 12th July, 2021

Matthias Deimel | Reka Marta Sabou

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Matthias Deimel, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 12. Juli 2021

Matthias Deimel

Acknowledgements

I want to thank my supervisor Dr. Marta Sabou MSc for supervising my thesis, as well as guiding me through the hard parts of the thesis. Further she provided me with the most valuable feedback on my work.

Further I want to thank Dr. Fajar Ekaputra for providing recommendations, which led to the optimal result of the thesis.

In the end I want to thank Mag. Peb Aryan for the help with the expCPS and the feedback on the implementations and the data structures.

Kurzfassung

In der derzeitigen technologischen Entwicklung ist es für Systeme wichtig die digitale und physische Welt miteinander zu verbinden. Aus diesem Grund wurden *Cyber Physical Systems (CPS)* eingeführt. Diese Systeme können Daten in der physischen Welt sammeln, in der digitalen Welt die korrekte weitere Vorgehensweise auswählen und diese Entscheidung zurück in die physische Welt schicken. Ein Beispiel für ein Cyber Physical System ist ein Smart Grid. Bei einem Smart Grid wird die Energie in einem Stromnetz so verteilt, dass jeder Benutzer die benötigte Menge an Energie erhält. Oft ist es wichtig den Entscheidungsprozess eines CPS zu verstehen, um die Ursache für auftretende Probleme zu finden oder den Entscheidungsprozess einem Benutzer zu erklären, um die Transparenz des Systems zu erhöhen. Die Erweiterung eines CPS, die es ermöglicht getroffene Entscheidungen zu erklären wird *explainable Cyber Physical System (expCPS)* genannt.

Ein expCPS basiert auf kausalen Beziehungen, zwischen Objekten in einem System (z.B.: Solarladestation, Gebäude, ...) oder Events (z.B.: fehlender Sonnenschein, weniger verfügbare Energie,..). Deshalb müssen die folgenden Probleme berücksichtig werden, wenn ein expCPS entwickelt wird: *(1) Welche Formalismen können verwendet werden um eine kausale Beziehung darzustellen* und *(2) wie ist es mögliche solche kausalen Beziehungen automatisiert zu finden?*. Derzeit gibt es viele verschiedene Ansätze in unterschiedlichen Feldern der Wissenschaft, aber in dieser Arbeit liegt der Fokus auf expCPS basierend auf Semantic Web Technologie und im speziellen in einer Smart Grid Umgebung.

Um diese Probleme zu lösen, benutzen wir die folgende Methodik: Am Anfang führen wir eine Literatursuche durch, um Möglichkeiten zu finden, Kausalität darzustellen, Kausalität automatisch zu sammeln und derzeitige Anwendungsfälle von Kausalität zu finden. Danach wählen wir aus den gefundenen Algorithmen die besten aus, um Kausalität zu finden, definieren Metriken, um den entsprechenden Algorithmus zu evaluieren und in einem Framework zu implementieren. Am Ende der Arbeit vergleichen wir die Implementierung der Algorithmen, um Kausalität zu finden, mithilfe der Metriken, die zuvor definiert wurden. Für diese Evaluierung werden "controlled experiments"verwendet. Zuerst wird jeder Algorithmus alleine evaluiert und anschließend werden diese Ergebnisse verglichen. Daher zeigt die Arbeit verschiedene Möglichkeiten, um Kausalität in einem expCPS zu repräsentieren und zu finden. Außerdem vergleicht die Arbeit verschiedene Anwendungsfälle für beide beschriebenen Probleme. Danach werden die optimalen Algo-

rithmen, um Kausalität zu finden in einem simulierten expCPS evaluiert, um Vor- und Nachteile aufzuzeigen. Diese Evaluierung findet in der BIFROST Simulations Engine statt.

Während der Arbeit leisten wir die folgenden Beiträge: Wir stellen eine Übersicht über wichtige Aspekte, die berücksichtigt werden müssen, wenn Kausalität dargestellt oder gefunden werden soll, vor. Außerdem entwickelt die Arbeit eine Definition von Kausalität. Die Definition beginnt mit einer Definition abgeleitet aus verschiedenen Definitionen, die derzeit in der Literatur verfügbar sind und wird in eine formale Problemstellung weiterentwickelt. Der Fokus der Implementierung liegt auf vier Algorithmen, um Kausalität zu finden: Granger Causality, Transfer Entropy (mit einem Kernel Estimator und einem Kraskov Estimator) und der Peter-Clark Algorithm. In der Evaluierung der Implementierung, zeigen wir, dass die Transfer Entropy mit einem Kraskov Estimator am besten performt.

Abstract

In the current technological uprise, it is important for systems to connect the physical and the digital world. For this purpose, *Cyber Physical Systems (CPS)* were introduced. These systems are able to collect data in the physical world, decide on the proper approach to solve problems in the digital world and communicate this result back to the physical world. An example for a Cyber Physical System is a smart grid, where the distribution of the energy in the grid is adjusted, based on the consumption of each consumer. Often it is important to understand the decision process of a CPS to either find the root cause of an appearing problem or explain the taken procedure to an end user to increase the transparency of the system. The extension of a CPS with the possibility to explain certain aspects of interest about the system, (e.g., unusual and possibly faulty states and behaviors) [GLV19] is called an *explainable Cyber Physical System (expCPS)*.

An expCPS is based on *causal relations* between either objects in the system (e.g., an electrical vehicle charging station, a building,...) or events happening during the operation of a system (e.g., reduced amount of available energy, lack of sunshine,...). Therefore, when designing and building an expCPS, the following topics need to be considered: (1) *which formalisms to use in order to represent causality relations* and (2) *how to automate the acquisition of such causality relation knowledge*? While several research fields have investigated these topics in very diverse settings and for several application domains, in this thesis we focus on which of these techniques can be applicable in the context of an expCPS System based on Semantic Web technologies and geared towards smart grid settings.

To that end, we follow the following methodology: In the start we perform an literature research to identify causality representation methods, causality acquisition algorithms and use cases of causal information in expCPS. Afterwards we select, the most suitable causality acquisition algorithms, define quality metrics to evaluate these algorithms and then implement these algorithms in a framework. In the end we compare the algorithms, based on the defined quality metrics, using controlled experiments. First the algorithms are evaluated on their own and then they are compared to each other.

In the thesis we make the following contributions: We provide an overview of important aspects to consider, while choosing or creating a causality representation and causality acquisition algorithm. Further, the thesis develops a definition of causality in the context of expCPS throughout the thesis. The definition starts with a collection of important

points described in the literature and is developed into a formal problem statement. Afterwards we propose a causality representation for an expCPS. The focus of the implementation is on four different causality acquisition algorithms: Granger Causality, Transfer Entropy (with a Kernel Estimator and a Kraskov Estimator) and the Peter-Clark Algorithm. In the evaluation of the implementation, we show the increased performance of the Transfer Entropy with a Kraskov Estimator compared to the other three algorithms.

Contents

CHAPTER 1

Introduction

1.1 Context and Motivating Scenario

In today's world *Cyber Physical Systems (CPS)* [Pla18, BG11] are on the uprise to support humans in their daily lives. Those systems can provide help in different areas of application. In [BG11] a CPS is definied as follows:

"The term cyber-physical systems (CPS) refers to a new generation of systems with integrated computational and physical capabilities that can interact with humans through many new modalities." [BG11]

An example for a Cyber Physical System is the smart electricity grid, where the system can monitor and redirect energy to stations, that have a high energy consumption. For a better understanding of these systems and to get the possibility to improve them, *explainable CPS (expCPS)* are introduced which can provide an explanation for unexpected system behaviors. An expCPS is defined as follows:

"The capability of both the system and its engineering tools to explain certain aspects of interest about the system, both in a human-comprehensible and machine-processable format." [GLV19]

To provide correct explanations, the explainable Cyber Physical System has to understand possible root causes for each feasible occurring event. This analysis is based on **causality**, where causality is a *relation between two objects or events, where one object or event causes another one.*

For example, a smart grid can be an expCPS, if the smart grid is able to provide explanations of occurring events to different users. The explanations provided by an

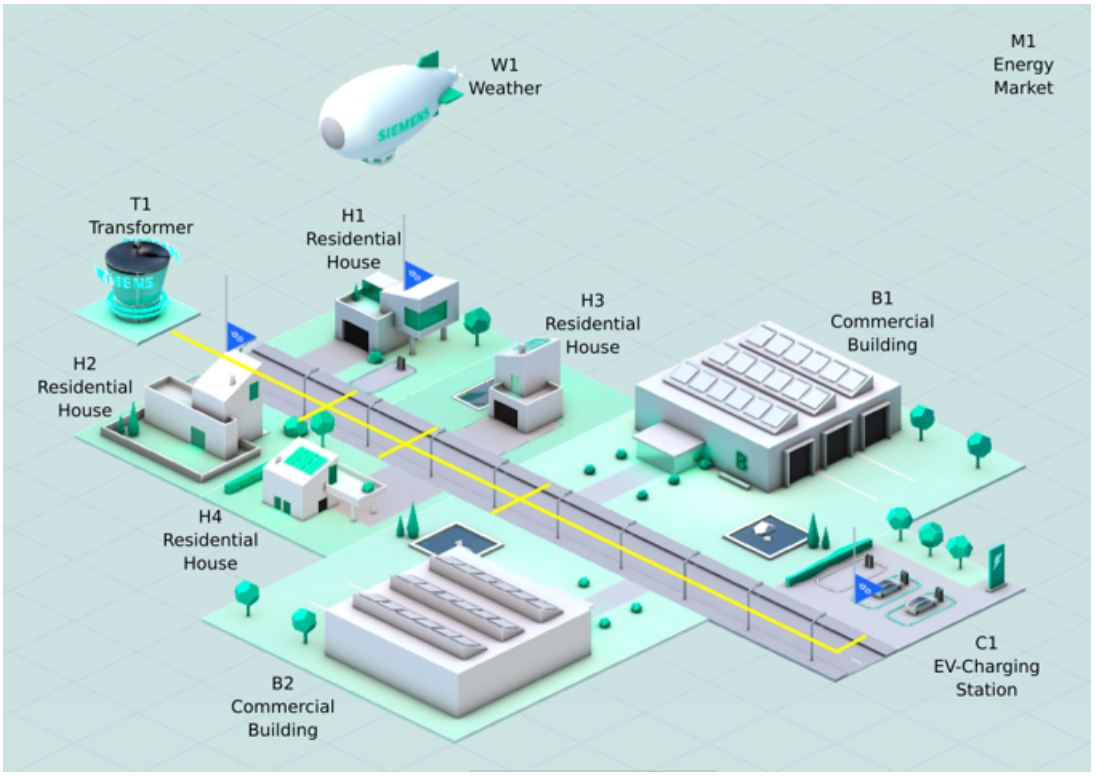


Figure 1.1: Example portion of Smart Grid in the BIFROST simulation engine.

expCPS increase the transparency of decisions, made by the smart grid. Therefore, multiple different users can benefit from these explanations. Imagine a scenario of a simple smart grid (Figure 1.1) with an electrical vehicle charging station (C1) with solar panels and multiple buildings (H1-H4, T1, B1, B2). In this example *a lack of sunshine leads to a lower energy production through the solar panels and this leads to a lack of energy in the smart grid, which then leads to a reduced amount of available energy for the end user*. The end user of the smart grid gains increased trust in the system, through the explanations, as the lack of energy is comprehensible and the time span of a lower charging speed is foreseeable. Further, an engineer knows there is no defect, and more sunshine will bring back the normal charging speed. If there is a defect the engineer can increase the safety and reliability of the smart grid, as he can find root causes of failures faster due to the explanations created by the expCPS.

The thesis aims to build a module for the expCPS shown in Figure 1.2 and described in [AES+20]. This system is built on top of the BIFROST simulation engine, where BIFROST [MDE+19] is a simulation engine, which is able to simulate a smart grid and provide simulated data about the smart grid. The expCPS shown in Figure 1.2 uses a set of different modules to acquire the data from BIFROST, to process this data and to provide explanations of selected events based on that data. The first

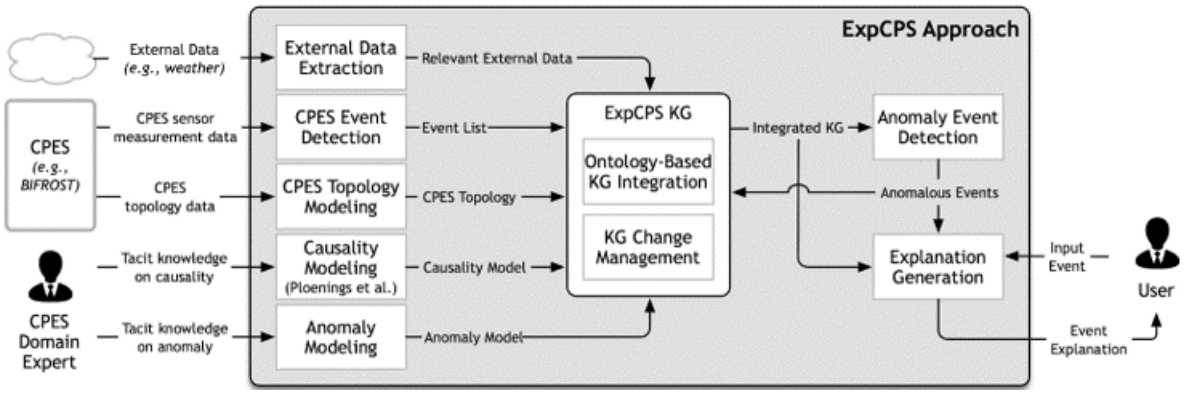


Figure 1.2: Structure of an expCPS, built on the BIFROST simulation engine, Reproduced from [AES+20]

module is the "CPES Topology Modeling", where the topology (e.g. residential buildings, transformers, power grid connectors, cabels,...) used to simulate a smart grid in the BIFROST simulation engine is acquired. Second, the module "CPES Event Detection" filters the recorded data to detect relevant events and third, the module "External Data Extraction" reads additional information (e.g. weather data). Further, the module "Anomaly Modeling" aims to acquire a set of possible anomalies from the simulation, while the module "Causality Modeling" acquires data to show the causal relations between the data collected by the remaining four modules.

After filtering the data from BIFROST, the expCPS builds and maintains a knowledge graph KG to save information about the system collected in the previous steps. This step is presented in the module "ExpCPS KG" and "Anomaly Event Detection" in Figure 1.2. The last component of the expCPS is the module "Explanation Generation", where the previously collected information in the knowledge graph is used to generate explanations of occurring events.

The module of the expCPS, the thesis is focusing on, is the module "Causality Modeling". The module built in the thesis should be able to acquire causality in the system from the data provided by BIFROST. This module collects the values of different objects provided in the simulation over time and uses these values to provide a set of causal relations in a representation, suitable for the other modules of the expCPS. An example for a causal relation is the relation between the amount of sunshine and the charging speed of a solar loading station, where a lack of sunshine decreases the charging speed of the solar loading station.

1.2 Problem statement

In explainable Cyber Physical Systems, it is critical to find and model causality relations in order to provide the root cause of different events. The extraction and modelling of

causality is a complex problem, which cannot be solved by connecting two events by a relation. For example, there is the possibility of an event occurring sometimes after another event has happened, but not every time. Another problem is the extraction of knowledge from systems to automate the process of acquiring causality information, as the goal is to require the least amount of expert knowledge as possible. Further, it should be possible to acquire and represent causality (semi-)automatically in an uniform applicable method to reach this goal. Therefore, the two main problems in this area are:

- **Research Problem 1: Causality representation:** Each type of CPS requires an optimal representation of causality, built from the information collected by the CPS, to provide an overview over each existing causal relation. Furthermore, for an optimal representation of causal relations in a system, it is necessary to take a lot of different aspects into account, as each representation aims to provide as much information as possible. A solution for this problem needs to contain every accessible information of a system built into a representation to present causal information.

- **Research Problem 2: Causality knowledge acquisition:** To build a causality representation as described in Research Problem 1, it is necessary to acquire the causality relations from a system. An example for a causality relation is the relation between the air conditioner and the temperature in a room, where a changing setting of the air conditioner is the cause for a changing temperature in the room. Furthermore, each system provides different information, different interfaces and different data models. Therefore, it is hard to automatically retrieve the causality relations in order to build the causality representation based on the information available.

Currently, there are multiple approaches ([SFSS09, PSL14, QDY+20, LÖC07]) from different domains to acquire and model causality. All these approaches use various techniques to either acquire, model, or acquire and model causality relations. There are also different approaches to evaluate gained causality knowledge ([QDY+20, LCH+19]) and different use cases to apply this knowledge([Gra69, CSK+18, Sch00]).

Despite the fact, that there are multiple systems, which are able to either acquire or represent causality in various research fields, there is currently no solution for an explainable Cyber Physical Smart Grid System. In this thesis we try to find an appropriate solution for the representation and acquisition in explainable Cyber Physical Smart Grid Systems.

Therefore, the goal of this thesis is to find an applicable approach to extract causality from an smart grid and model all causality relations in an appropriate way. The thesis further applies the chosen approach to a simulated example of an expCPS, hosted as module in the BIFROST simulation engine. The BIFROST simulation engine can be seen in Figure 1.1.

1.3 Aims

To successfully build a module in the expCPS, the thesis has to answer multiple research questions (RQ). These research questions are:

- **RQ1:** *What are classical ways to represent causality information and which formalisms and notions are used to represent causality information?*

  At the start, the research aims to go beyond literature about Cyber Physical Systems in order to provide an overview of current solutions from other fields (e.g.: linguistics or economics) to represent causality information. Further the thesis wants to show an overview over existing formalisms, notions and methods to represent causality information.

  After the initial overview, this Research Question aims to summarize these approaches and extract the crucial aspects which are required to model causality information. Furthermore, the thesis contains an overview of important aspects and the description, why they are important for a causality representation. Accordingly, it shows a more in-depth overview of approaches to solve Research Problem 1 and shows important aspects to consider in order to solve Research Problem 2.

- **RQ2:** *Which ways are already used to acquire the information required to build a causality representation and how these approaches use the acquired information to build a causality representation?*

  In this part of the research, the knowledge from the previous question is applied in the context of a concrete explainable CPS. In the first step of this Research Question, the thesis aims to provide current solutions of other (explainable) Cyber Physical Systems and how these systems acquire and model causality informations. Afterwards the thesis uses the previous literature study to propose a causality representation, which can be used as a basis for an explainable Cyber Physical Smart Grid System.

  These results aim to provide a proposal to solve Research Problem 1, which can be used to improve an expCPS System. Further this Research Question aims to provide an overview over different import aspects of the application of the different causality representation methods.

- **RQ3:** *What approaches are already used to acquire causality information from a system and is it possible to apply these approaches to an expCPS?*

  This Research Question aims to analyze already existing approaches from literature to acquire causality from a system. Further, this part of the research shows the applicability of the identified approaches to an explainable Cyber Physical System.

  The results of the literature study aim to show important aspects of these methods, which have to be considered before choosing a method to use in an expCPS. These results also provide an overview of methods, which must be considered for use in

the BIFROST simulation engine. Therefore, this Research Question provides an overview of methods to solve Research Problem 2.

- **RQ4:** *How can causality information, with the help of the approaches from RQ3, be acquired from the BIFROST simulation engine? What are advantages and disadvantages of each method?*

  After creating an overview of possible causality acquisition algorithms for explainable Cyber Physical Smart Grid Systems, these methods are applied to a concrete simulation engine, the BIFROST engine developed by Siemens. The most promising causality acquisition algorithms proposed in the last question are chosen and applied to the BIFROST engine and implemented to acquire causality information from the BIFROST modules. Further, the implemented causality acquisition algorithms and the extracted causality information shall be evaluated. The evaluation has to focus on the quality of the results and the runtime of the algorithm in the simulation engine.

  During the development and evaluation of this Research Question, a solution to Research Problem 2 should be provided.

1.4 Methodology

In the thesis the following methodology is used to create the contributions mentioned in Section 1.5 and to solve the Research Questions mentioned in Section 1.3. An overview over the methodology is shown in Figure 1.3.

Literature study

In the first step of the thesis, an extensive literature research is performed. The literature study aims to provide information over different subjects, required for the thesis. These subjects are:

- existing causality representation methods
- existing causality acquisition algorithms
- use cases of causal information in expCPS

To collect this information, current literature is studied to provide the required knowledge. Even though the literature study aims to answer RQ1 to RQ3, it underpins the work of the entire thesis.

Algorithm Selection & Implementation

After the literature study, we aim to choose the most suitable approaches to acquire causality information from the BIFROST simulation engine and build a module using these approaches/algorithms to acquire the causal relations from the data provided by

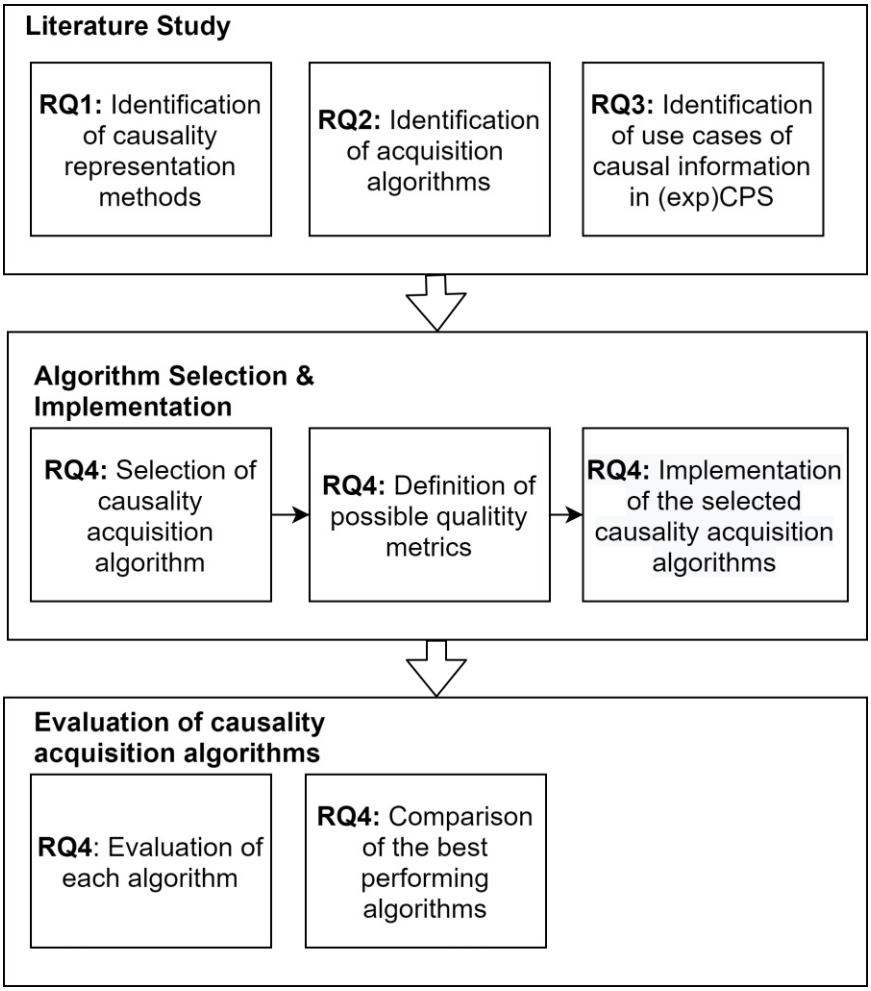


Figure 1.3: Thesis methodology overview.

the simulated environment of BIFROST. After successfully implementing the algorithm, a set of possible quality metrics has to be defined and implemented to be able to evaluate the different algorithms. Finally the implementation should be able to provide a set of causal relations existing in the BIFROST simulation engine. The implementation addresses Research Question 4 (RQ4).

Evaluation of Causality Acquisition Algorithms

In the last step, the thesis provides an evaluation, using controlled experiments (presented in [WRH+12]) of the implemented causality acquisition algorithms. For this evaluation, the data acquired from the BIFROST simulation engine, will be used. The evaluation will analyse the behavior of different causality acquisition algorithms. In the first step the evaluation assesses each algorithm on its own and shows advantages and disadvantages of the algorithms chosen during the selection & implementation step. Afterwards the

evaluation compares the results of the different algorithms. In this final step Research Question 4 is addressed.

1.5 Contributions

The thesis provides the following contributions to the development of a causality module for the expCPS module in the BIFROST simulation engine by answering the research questions:

- **Definition of causality in the context of expCPS**, starting with an overarching working definition of causality for expCPS based on an overview of definitions of causality used in various research fields (Section 2.1) and a formalised definition in Section 4.1
- **Important aspects to consider, when creating or selecting a causality representation**, based on a list of current possibilities, provided in the literature to represent causality (Section 2.3), we acquire an overview over important aspects (Section 2.4), which need to be considered to build a representation with each information required to learn causal information. This contribution discusses RQ1.
- **A proposal for a causality representation for expCPS**, derived from a comparison of different expCPS (Section 3.2), which use a broad range of causality representations and causality acquisition algorithms and taking into account the important aspects to consider, when creating or selecting a causality representation from Section 2.4. This contribution is related to RQ2.
- **The identification of important aspects to consider, when creating or selecting a causality acquisition algorithm and the applicability of these algorithms to the BIFROST simulation engine**, based on an overview of causality acquisition algorithms in Section 4.2, we present a set of import aspects of causality acquisition algorithms (Section 3.2)as response to RQ3.
- **An evaluation and comparison of different causality acquisition algorithms**, based on timeseries acquired from the BIFROST simulation engine, as a contribution to RQ4. This evaluation is presented in Chapter 6.

1.6 Structure of the Work

The thesis is structured as follows: Chapter 2 introduces causality representations, where a definition of causality, a list of possibilities to represent causal relations and their advantages and disadvantages are discussed to address RQ1. Afterwards, Chapter 3 details different application areas of the causality representations from Chapter 2, related to RQ2. It also points out different aspects to consider when choosing the correct causality representation. In Chapter 4 different causality acquisition algorithms are

presented, where the chapter starts with a detailed problem statement and continues with a presentation of different causality acquisition algorithms to discuss RQ3. Chapter 4 also shows the applicability of the algorithms to the BIFROST simulation engine. The thesis continues with Chapter 5, where the implementation framework to evaluate the causality acquisition algorithms is presented. Afterwards the thesis continues with an evaluation of the algorithms implemented in the framework in Chapter 6. Chapter 5 and 6 aim to solve RQ4. Finally Chapter 7 concludes the thesis by recapping the Research Questions, providing a list of limitations and assumptions of the thesis and present possible future work.

CHAPTER 2

Causality Representation

This chapter investigates RQ1 in terms of the notion of causality and various approaches to represent it. In Section 2.1 we overview definitions of causality drawn from various research fields and distill an overarching working definition for ExpCPS, which is the first contribution of the thesis. Accordingly, based on this definition an exemplifying scenario is introduced (Section 2.2) as a key instrument to ground the notions of the definition and to serve for concretely illustrating the causality representation approaches reviewed. Section 2.3 briefly describes 10 different approaches to causality modeling and exemplifies them by instantiating them in the context of the scenario defined in Section 2.2. Section 2.4 concludes this chapter with an overview of the main features of causality representations which serve as a means to compare the various representation approaches reviewed. This comparison is another contribution of the thesis.

2.1 Defining Causality

This section of the thesis gives an overview over current definitions of causality from different research areas. After the initial overview, another definition is shown, based and extended from previous ones. The following list shows different definitions from current literature:

- "Causality is one of the main semantic relationships between events where an event (CAUSE) results in another event (EFFECT) to happen or hold." [MGC+16]
- "Causality is the relationship that the state of one property is determined by the state of another property. This relationship may be a physical process between two physical properties or, more generally, by relating a physical property to socio-psychological property such as amount of rage during a blackout. By modeling it in more abstract term than a physical process, it allows for a relationship to also

be explainable in terms of how the knowledge was collected (e.g., explicitly from expert of a well-founded discipline or some recent studies).” [AES+20]

- “Causal relationships require the modeling of causes and effects and should support the integration and use of different causal theories as discussed. Correlation refers to two events that have a common cause “ [SFSS09]
- “Causality describes the cause-effect relationship between changes of process variables” [YDSC14]
- “Causality (also referred to as causation) is the relation between an event (the cause) and a second event (the effect), where the second event is understood as a consequence of the first. In common usage, causality is also the relation between a set of factors (causes) and a phenomenon (the effect). Anything that affects an effect is a factor of that effect. A direct factor is a factor that affects an effect directly, that is, without any intervening factors. The connection between a cause(s) and an effect in this way can also be referred to as a causal nexus. Though the causes and effects are typically related to changes or events, candidates include objects, processes, properties, variables, facts, and states of affairs.” [YDSC14]
- “X could be termed as to ‘cause’ Y if the predictability of Y is improved by incorporating information about X” [YDSC14]
- “We say that x(k) is causing y(k) if we are better able to predict y(k) using all available information than if the information apart from x(k) had been used” [YDSC14]
- “We understand causation to be a relation between particular events: something happens and causes something else to happen. Each cause is a particular event and each effect is a particular event. An event A can have more than one cause, none of which alone suffice to produce A. An event A can also be overdetermined: it can have more than one set of causes that suffice for A to occur. We assume that causation is transitive, irreflexive, and antisymmetric.”[Maz09]

Based on this list and further research, the following definition was created. It will be used for a more in depth understanding of the following methods and formalities to represent causality. Further this definition will be applied to examples of these methods and formalities:

“Causality is the relation between group A (causes) and group B (effects) of entities, where the second group is understood as a consequence of the first one. An entity can be either an object in a system or an event, which appears during the operation of a system. Further each group must have at least one entity. Each group can have multiple effects and causes. Additionally, causality is generic if an entity is an object which has a causality relation with another object and concrete if each entity is an event, where an event is the cause of another event.”

has a causality relation with	weather	engineer	solar loading station	electrical loading station	car
weather	X	no	yes	no	no
engineer	no	X	no	yes	no
solar loading station	no	no	X	no	no
electrical loading station	no	no	yes	X	no
car	no	no	yes	no	X

Table 2.1: Generic Causality in the exemplifying scenario

2.2 Exemplifying scenario

The following example describes a scenario, which is used in the following section to demonstrate different approaches to represent generic and concrete causality. The focus of the example is a solar car loading station, which is influenced by the weather. Good weather (where good weather is sunny weather) increases the loading speed of the loading station and bad weather decreases the loading speed. Further the loading station influences a loading car, where a low loading speed increases the loading time of the car and a high loading speed decreases the loading time. Additionally, the loading level of the car influences the loading level of the loading station, as a high loading level in the car yields a lower loss of energy than a low level in the car. As last entity, an electric loading station is introduced, which can be activated or deactivated by an engineer. If the electric loading station is active, the weather does not matter, as the car will always use the electric loading station instead of the solar one. Due to this construction, the electric loading station influences the loading time of the car, as it keeps it consistent on a high level.
Applied to our definition, we can derive two representations of causality relation, a generic one and a concrete one. In Figure 2.1 and Table 2.1 we can see the generic representation, where the weather and the engineer are the root causes, which effect the solar loading station or the electrical loading station, while both, the electrical and the solar loading station influence the car. As last relation in the generic representation, we can see the car influence the solar loading station. In Figure 2.2 and Table 2.2 is a concrete causality representation of the solar loading station modeled. The root causes are different events, based on the objects of the generic representation. Therefore, if the electrical loading station is active or deactivated, it increases or lowers the loading time of the car. Further, if the weather changes from sun to rain or the other way around the loading speed of the solar loading station changes accordingly. As soon as the loading speed of the solar loading station changes, the loading time of the car changes as well. In the second graph in Figure 2.2 we see the difference in the loading level of the car changes the loading level of the solar loading station.

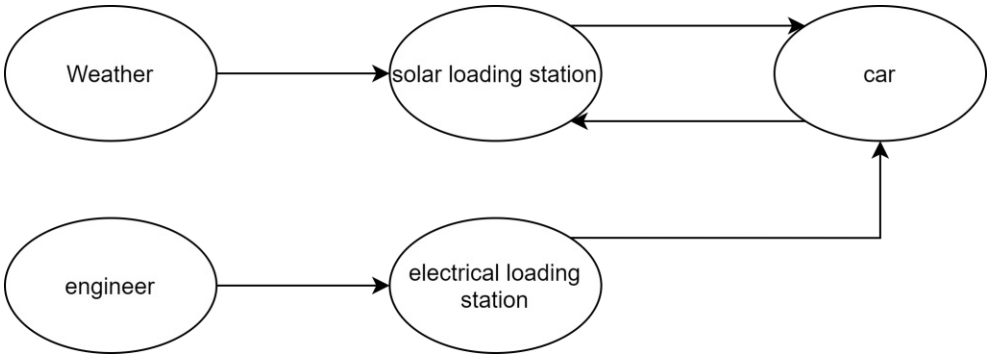


Figure 2.1: Generic representation of the exemplifying scenario

influences	sun/rain	high/low loading speed	electrical loading station active/inactive	low/high loading time of the car	low/high loading level of the car	low/high level of the solar loading station
sun/rain	X	yes	no	no	no	no
high/low loading speed	no	X	no	no	no	no
electrical loading station active/inactive	no	no	X	yes	no	no
low/high loading time of the car	no	yes	no	X	no	no
low/high loading level of the car	no	yes	no	no	X	yes
low/high level of the solar loading station	no	yes	no	no	yes	X

Table 2.2: Concrete causality in the exemplifying scenario

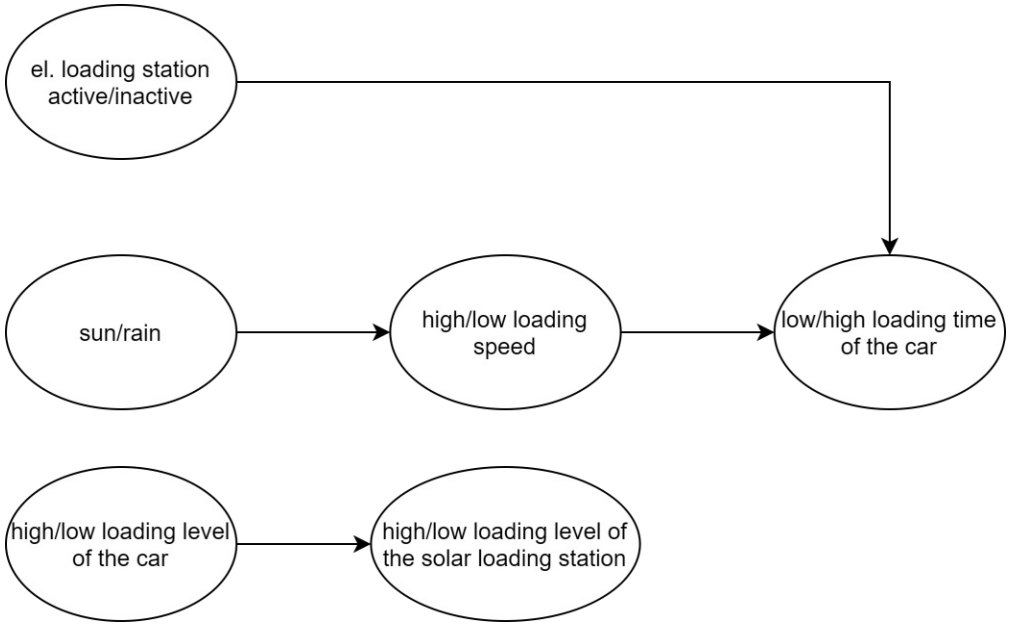


Figure 2.2: Concrete representation of the exemplifying scenario

2.3 Methods for causality representation

This section presents different methods and formalities to represent causality. Each method is presented with a short introduction and afterwards applied to the loading station example.

2.3.1 Nature Language Processing

There are multiple approaches to find and annotate events and relations. The goal of these approaches is to automatically detect and represent these events and relations. In the following section these approaches are presented with a special focus on different kinds of relations and events considering causality.

Causal and Temporal Relation Scheme (CaTeRS)

In this approach, presented in [MGC+16], the goal is to connect two events with either a temporal or causal relation and therefore, a connected network of events is created. The paper presents three major causal relations (CAUSE, ENABLE, PREVENT), which in conjunction with a verb (before, overlaps, during), represent the timeline of the events. A cause relation represents a causality relation where event A always implies event B. The enable keyword implies, if event A has happened, event B can happen, but it does not have to happen. If event A does not happen, it is not possible for event B to happen. Further a prevent relation represents a relation, where after event A happened, it is no longer possible for event B to happen.

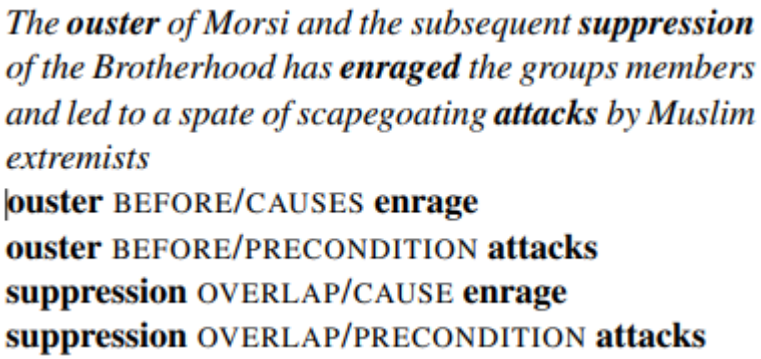


Figure 2.3: Example of the annotation method RED, Reproduced from [OWBP16]

The paper also introduces a differentiation between different types of events. All events are split into the categories: Event-of-state, Event-of-change, Event-type, Physical-condition, Occurring and Natural-phenomenon. According to the paper, it is possible to objectively classify events with these different types of events.

RED

Similar to the last approach, this one, proposed in [OWBP16], tries to find, relate and connect events. RED also uses the causality relation types "CAUSE" and "PRECONDITION", where "PRECONDITION" is the same as "ENABLE" in the CaTeRS-Scheme. The difference between these two methods is the use of "PREVENT" in CaTeRS, where RED uses a negation of an event to represent this type of relation. Further explained this means, if event A prevents event B, we can also say, event A is a precondition to the negation of event B. The advantage of the removal of the "PREVENT" relation is an easier way to annotate relations, but the disadvantage is the inability to present "PREVENT" for events with more than one state. RED also uses these types in conjunction with another verb to describe the timeline of the events. In Figure 2.3 we can see all possible combinations of relation types and timeline verbs.

CLINK

CLINK is a basic approach, presented in [MSTS14], to solve the same issues as the previous two methods. It describes two different types of events, the cause and the effect. Those two events are connected through a causality relation, which is one of three types: CAUSE, ENABLE and PREVENT. They work the same as previous approaches. The approach further adds a negation (NEG) to each attribute to enable a representation of complex stories.

BECauSE 2.0

In this approach, definied in [DLC17], multiple different types of causality are used to represent causality. Like the previous method, this one also connects two events with each other, but it uses different types of relations to represent these. The used relations are: CONSEQUENCE, MOTIVATION and PURPOSE. A CONSEQUENCE describes the same relation as a CAUSE in the previous methods. MOTIVATION represents a relation, where an actor experiences the cause and based on the cause thinks about an action or takes an action, which is the effect of the experience. At last, a PURPOSE relation is a relation, where the actor chooses the effect out of multiple possible options before the cause happened, to make a specific cause happen.

Example

We conclude this section about representation of causality in nature language processing with an example of CaTeRS applied to our running example, as it is quite similar to CLINK and RED and it is not possible to apply BECauSE 2.0 to the example, as the relations are not designed for an example like this. To be able to apply CaTeRS to our example, we use different events from our example with relations from the method. We can not represent a generic causality with any linguistic approach as these approaches work with events. The following relations can be deducted from our example:

- (sun) CAUSE DURING (high loading speed of the loading station)
- (rain) CAUSE OVERLAPS (low loading speed of the loading station)
- (high loading speed of the loading station) ENABLE BEFORE (low loading time of the car)
- (low loading speed of the loading station) CAUSE BEFORE (high loading time of the car)
- (low loading level of the car) PREVENTS BEFORE (high loading level of the loading station)
- (high loading level of the car) ENABLE BEFORE (high loading level of the loading station)

In the example we can see the event “sun” causes a high loading speed of the loading station, which is the rule in our example. Further, the loading speed only increases while the sun is shining. We could also add another relation where weather is only sunny before the loading of the car, which would also cause a high loading speed of the car. Additionally, we see the loading speed has to be high before the car arrives, as it does not change during the loading process. At the end we can also see a low loading level of the car prevents a high loading level of the loading station, as the car needs more energy

to be fully filled again. It is not possible to model the engineer with this approach as we have no way to model a conditional causality.

Finally, we can see most of these NLP approaches provide two important aspects for the representation of causality. First we can see the use of verbs to set cause and effect into a chronological relation. In these approaches the used keywords are BEFORE, DURING, OVERLAPS or slight variations. The second import aspect is the differentiation of causality relations. In these representations we can clearly see how the cause is related to the effect, as we can see, if the effect can, cannot or has to happen after the cause.

In the end we can summarize, these approaches are not fitting for our example, as they are unable to represent conditional causality, generic causality or weighted causality. Further we can apply these approaches only in a limited way to our example, as it is a big system instead of a text.

2.3.2 Adjacency- & Reachability Matrix

Adjacency- & Reachability matrices are matrices (presented in [YDSC14]), where each object of a system is listed. A “1” in the matrix represents a causality relation between these two objects, a ”0” indicates no causality relation. An adjacency matrix represents only direct causality relations, where a reachability matrix also represents indirect causality relations. In Figure 2.4 the difference between these two types of matrices is clearly stated, where X4 has a direct relation with X2, but not with X3, as the water in the example first flows into tank X2 and from there into X3. In the adjacency matrix the relation between X4 and X3 is not modeled, in the reachability matrix it is.

It is also possible to transform an adjacency matrix into a reachability matrix by adding every indirect causality relation. This can be achieved by applying the transitivity rule to every causality relation. Therefore, for each relation where A influences B and B influences C, also add an entry for A influences C.

Applied to our example, we can model a generic causality relation with each matrix (modelled in Table 2.3 and 2.4). If we compare both matrices, we see the influence of the engineer on the car is only modeled in the reachability matrix, as it is an indirect relation, where the engineer influences the electrical loading station, which influences the car.

This approach can be used to represent different generic causality relations, but is not able to present a concrete relation due to the fact, that it represents objects and not events. It can further show the transitivity of causality. These matrices are ideal to present an overview over existing causality relations.

2.3.3 Signed Directed Graph (SDG)

Similar to the adjacency or reachability matrix (presented in [YDSC14]), it is possible to model causality relations in a Signed Directed Graph. Each object is presented as a node in the graph, where a relation is a directed arrow pointing from the effect to the cause. In this representation it is further possible to represent negative influences, where an

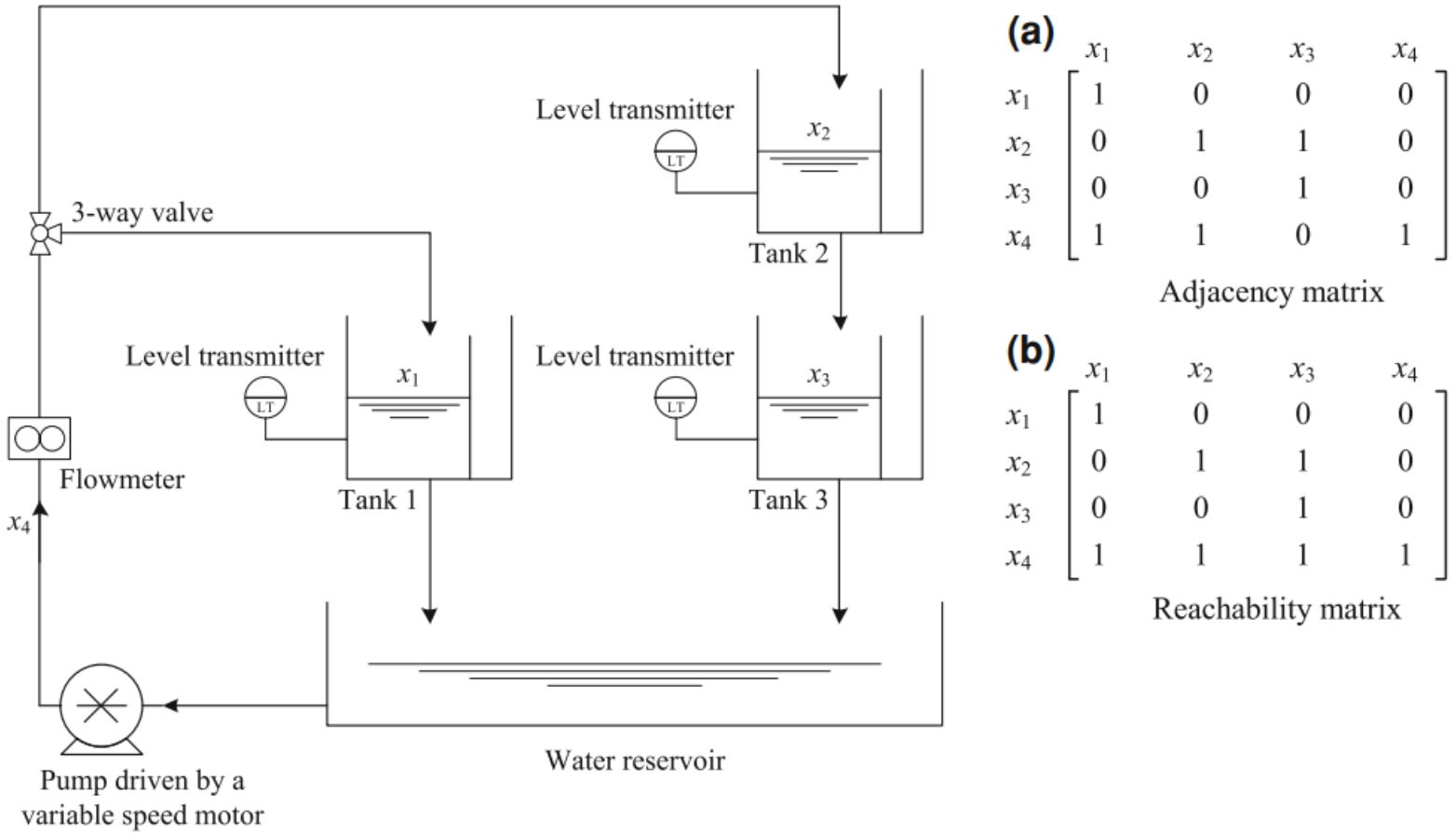


Figure 2.4: Example of a Adjacency- & Reachability Matrix, Reproduced from [YDSC14]

	Weather	Engineer	el. Loading Station	Solar Loading Station	Car
Weather	1	0	0	1	0
Engineer	0	1	1	0	0
el. Loading Station	0	0	1	0	1
Solar Loading Station	0	0	0	1	1
Car	0	0	0	1	1

Table 2.3: Adjacency matrix for the solar loading station example

	Weather	Engineer	el. Loading Station	Solar Loading Station	Car
Weather	1	0	0	1	1
Engineer	0	1	1	0	1
el. Loading Station	0	0	1	0	1
Solar Loading Station	0	0	0	1	1
Car	0	0	0	1	1

Table 2.4: Reachability matrix for the solar loading station example

event in object A has a negative effect in object B. Negative causalities are represented with a dashed line. Therefore, if we take the tank example in Figure 2.5, where F1 is the flow into the tank, F2 is the flow out of the tank and L is the level of the liquid in the tank, we can see a positive causality relation between the input, output and level of the liquid, as more flow into the tank increases the level, which increases the flow out of the tank. Further we see a negative relation between the outflow and the level, as more liquid leaves the tank, less liquid is in the tank.

According to the paper, this approach can represent a generic causality relation. Applied to the running example with the solar loading station, the representation can be seen in Figure 2.6. In this figure we see the weather has a positive impact on the loading station, as good weather implies a high loading speed of the loading station. We further see a negative relation between the car and the solar loading station, as a low energy level in the car causes a lower energy level in the solar loading station than a high energy level of

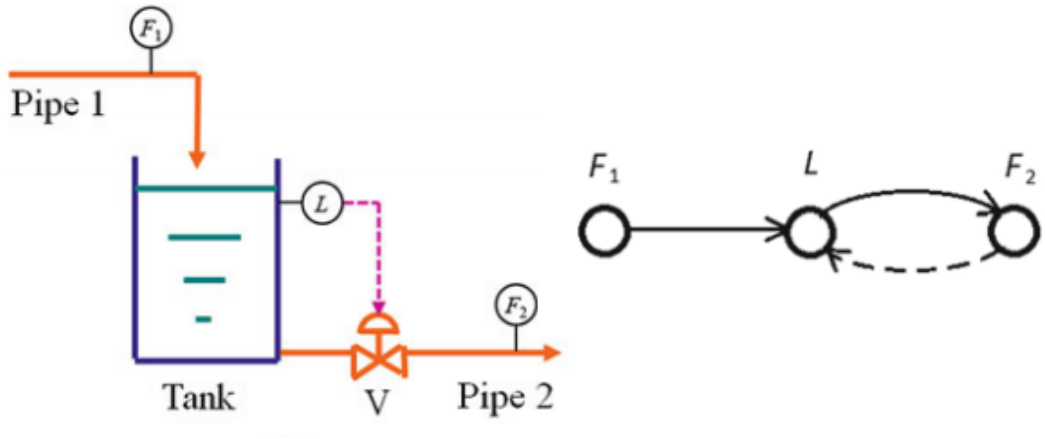


Figure 2.5: Example for a SDG, Reproduced from [YDSC14]

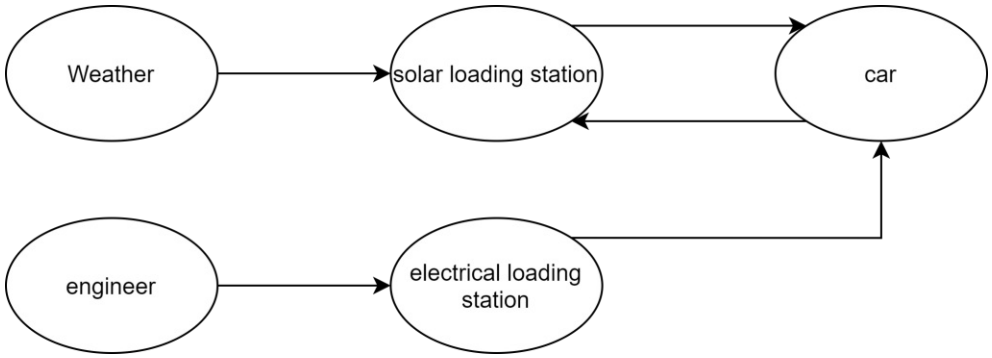


Figure 2.6: Signed Directed Graph for the solar loading station example

the car.

This approach is based on the matrices mentioned before and therefore has similar advantages and disadvantages. It can represent generic causality and additionally is able to present binary weights. Even though it is possible to build a similar graph with events, it is not possible to decide positive and negative relations for these events. The decision, if a relation is positive or negative, is often quite subjective, as someone could consider rain as good weather and sun as bad weather.

2.3.4 Matrix Layout Plot

This plot, defined in [YDSC14], presents the influence between two objects. Therefore, on the top are always the cause variables, while all effect variables are on the left. The representation shows the influence of one object to another. In Paper [YDSC14] the information flow is measured by a method called partial directed coherence (PDC), which can provide a normalized function of the information flow. A more detailed information can be found in Paper [BS01]. Further the plot can only show direct causality

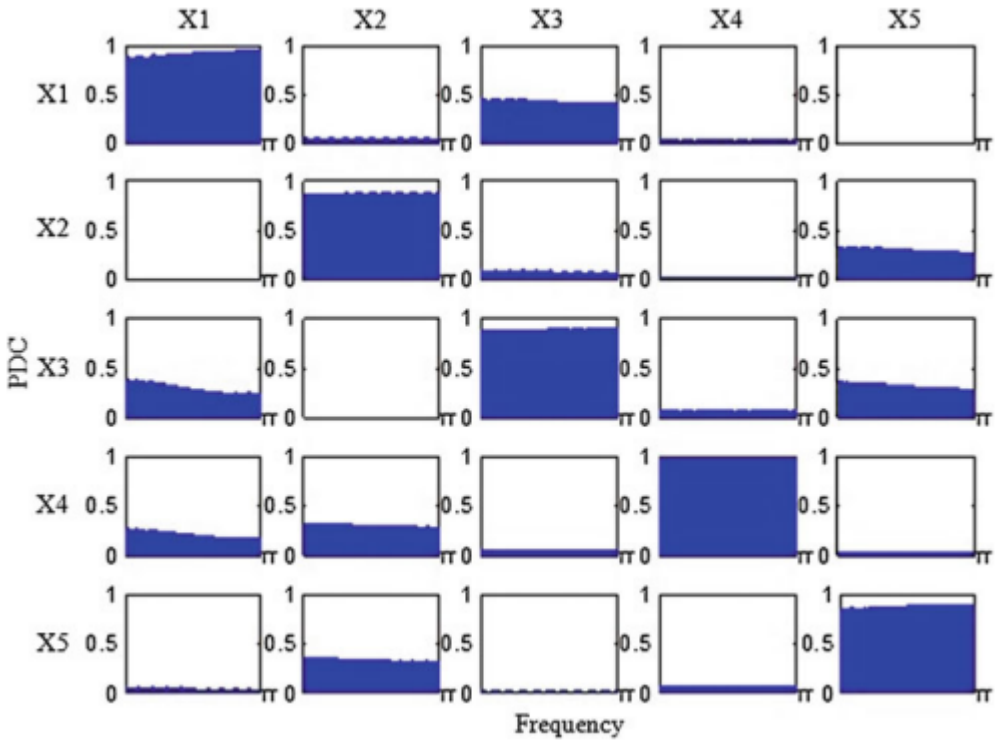


Figure 2.7: Example of a Matrix Layout Plot, Reproduced from [YDSC14]

relations. Therefore, we can see in the plot in Figure 2.7 an example, where X3 has a high information flow towards X1 and a low information flow towards X2.

2.3.5 Influence Diagram in Canonical Form

This representation, presented in [HS13], is an acyclic directed graph with two different types of nodes to represent causality relations. The two types are decision nodes and chance nodes, where a decision node represents a decision chosen by an actor, while a chance node is a result, based on a previous decision. Additionally, the graph contains deterministic nodes, which is a chance node, based on deterministic functions. Therefore, each time a decision before a deterministic chance node has the same decision, the chance node will also have the same state. To ensure this consistency, the diagram further introduces mapping variables, which map the state of a previous decision node onto a chance node. In the graph each decision node is represented by a square, a chance node is represented by a circle and deterministic chance nodes are represented by a doubled circle.

In Figure 2.8 an example, presented in Paper [HS13], is shown. The example is about a medical treatment, where the doctor either chooses to give a recommendation (r) to take the medical treatment or not. Based on this recommendation the patient either takes

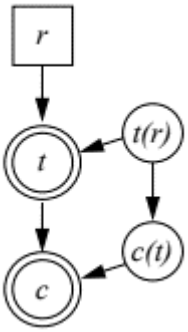


Figure 2.8: Example of the Influence Diagram in Canonical Form, Reproduced from [HS13]

(t) the treatment or not. The doctor can not influence this decision and therefore it is modelled as a chance variable. As last node in the graph we can see, if the patient is cured (c). To ensure consistency there are two mapping variables t(r) and c(t), which declare, if the patient takes the treatment based on the recommendation and if the patient is cured based on the decision if he took the treatment.

For our running example, seen in Figure 2.9, we use dashed circles instead of double lined circles to represent deterministic nodes and we furthermore use the following mapping:

- w ... weather
- e ... engineer
- el ... loading speed of the electrical loading station
- so ... loading speed of solar loading station
- c ... loading time of the car
- ll ... loading level of the solar loading station
- lc ... loading level of the car

In the running example we can see the engineer, the weather and the loading level of the car as decision nodes. Even though the weather and the loading level are not typical actors, they are the base for further chance nodes. We could introduce another person, which decides the moment to load the car, which would overtake the position of a decision node. We can further see four mapping variables, where three are simply based on the previous decision node. The last one is c(el,w), which presents the mapping from the electrical loading station (active/inactive) and the solar loading station (slow/fast loading speed) to the loading time of the car.

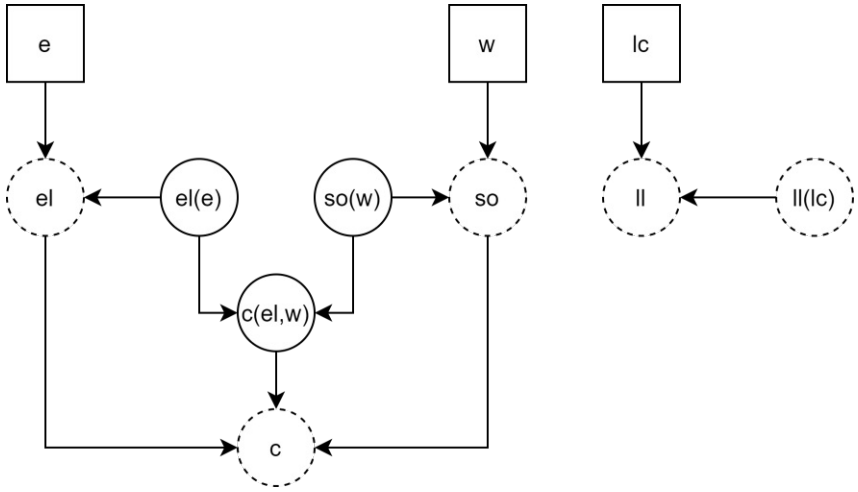


Figure 2.9: Influence Diagram in Canonical Form for the solar loading station example

This approach represents concrete causality, as it shows different states of properties of the objects, which where previously modeled as events. It can model a lot of different states in a compact view, which further ensures consistency.

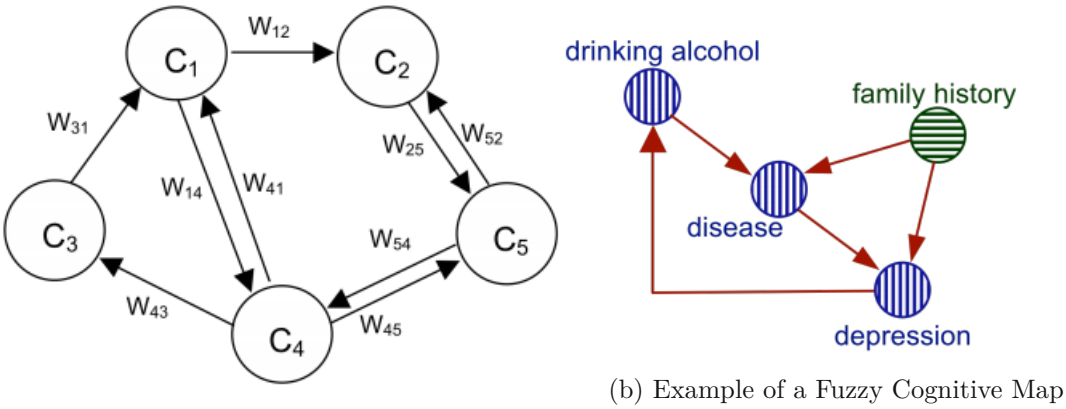
2.3.6 Fuzzy Cognitive Map (FCM)

A FCM, proposed by [Maz09], is represented by using concepts, where each concept either represents a state or a characteristic. In typical cases this means, each concept is an event or a trend of the system the FCM is modeled for. To create a FCM each concept relates to other concepts through a causality relation, which is represented through an arrow. Each relation is further weighted to represent the strength of the relation. In Figure 2.10a we can see how a formal representation can look like. We further see the possibility to build cyclic dependencies. An example for cyclic dependencies can be seen in Figure 2.10b, where we can see the consumption of alcohol leads to disease, which leads to depression, which again leads to the consumption of alcohol.

The FCM is built to represent events, therefore we can use it to build our example with concrete causality. In Figure 2.11 a FCM is built according to our running example. In this representation, we can see that even indirect causalities are modeled, as the weighting is able to provide a differentiation between the different causality relations. In comparison with previous representations, this representation does not represent negative causality relations and has no ability to provide an additional description.

2.3.7 Extended SOSA Approach (Sensor, Observation, Sample, Actuation)

This approach, presented in Paper [AES+20], uses multiple different components to represent a causality relation. In the focus of the representation are so called “features



(a) Structure of a Fuzzy Cognitive Map

(b) Example of a Fuzzy Cognitive Map

Figure 2.10: Fuzzy Cognitive Map, Reproduced from [Maz09]

of interest", which model an observable component. Each of the features are connected with properties, which are further connected between each other to represent different relations. One possible relation is the causality relation. In Figure 2.12 a similar example to our own example is presented, where the weather influences the photovoltaic system, which influences the transformer, which further influences a solar loading station.

We do not model our own example for this approach as Figure 2.12 already presents a similar example with all relevant information. This approach demonstrates a high-level representation of causality. It only shows a generic relation between two properties of two components and demonstrates a single type of causality relation.

2.3.8 Dynamic Uncertain Causality Graph (DUCG)

The Dynamic Uncertain Causality Graph (presented in [Zha12]) is an extension of the DUG. The DUG represents causes and effects, where an effect can be the cause of another effect. Further there is always a root cause, which is not an effect of another modelled cause. In the DUG it is also possible to use logic gates to unite multiple possible causes for an effect. Therefore, it is possible for an effect to have different sets of causes and preconditions. Each basic root is modeled as a square, while an effect is modelled as a circle and a causality relation as an arrow from cause to effect. In Figure 2.13 we can see B1 and B5 as root causes, X2 (where the previous cause of X2 is not modeled) and X4 as effects and G3 as a gate to combine multiple causes.

The two extensions from DUG to DUCG are a Conditional Linkage Event and a Default Event. The Conditional Linkage Event represents a causality relation, which only holds under certain terms (modeled with dashed arrows), while the Default Event represents missing information. In the example in [fig example DUCG] we see an example of an alarm system, where either an earthquake, an intruder or a rat can trigger the alarm. There are two different types of trigger mechanisms, infrared and vibration. An earthquake can

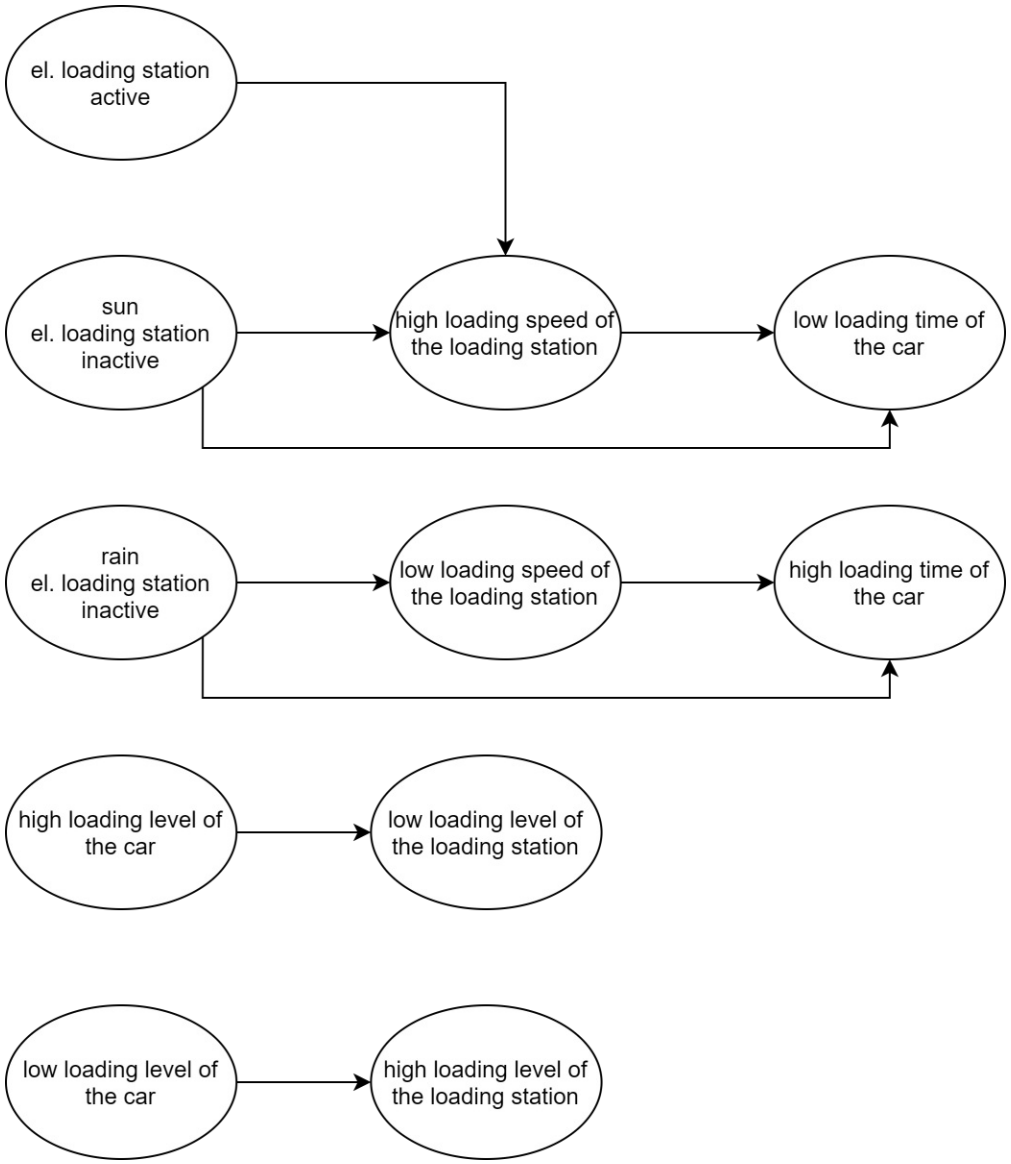


Figure 2.11: Fuzzy Cognitive Map for the solar loading station example

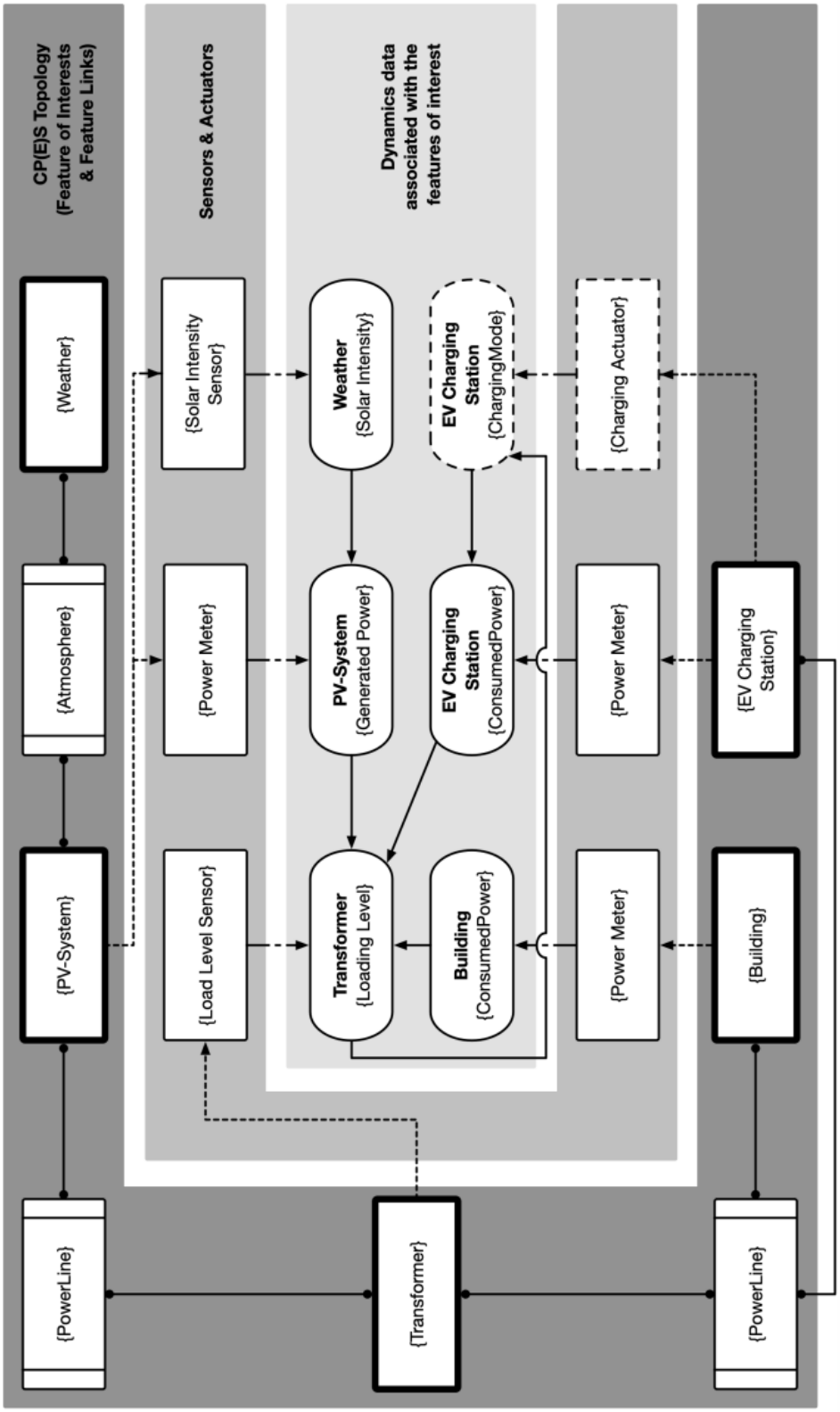


Figure 2.12: Example for the SOSA Approach, Reproduced from [AES+20]

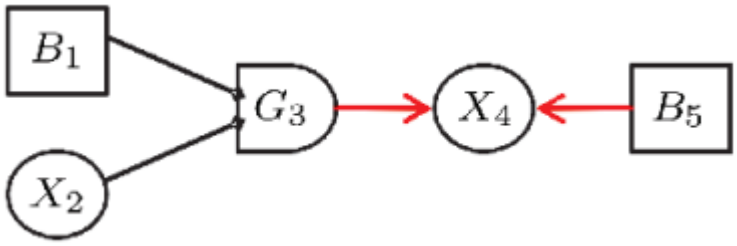


Figure 2.13: Structure of a Dynamic Uncertain Graph

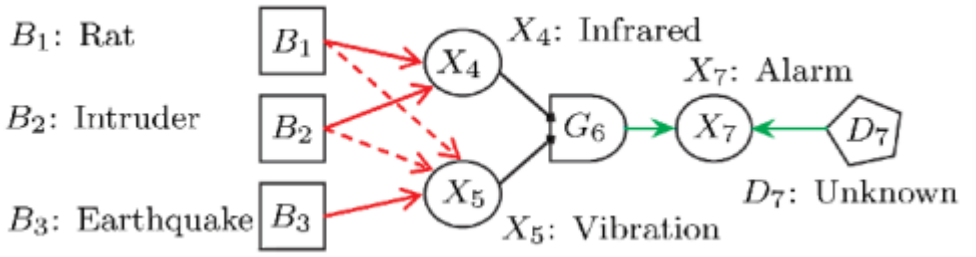


Figure 2.14: Example of a Dynamic Uncertain Causality Graph, Reproduced from [Zha12]

only trigger the vibration alarm, while the intruder and the rat can trigger both. We can further see in the graph, if an earthquake appears, the vibration alarm can not be triggered by an intruder or rat as the earthquake causes stronger vibrations. The same principle is valid for the infrared alarm as it can not be triggered by the rat if an intruder already triggers it. As there can be more possible events to activate the alarm, there is also a Default Event (D7) to model unknown events.

The running example according to the DUCG representation, is modeled in Figure 2.15, where the root events are the weather and the decision of the engineer. We further see a conditional relation between the loading of the solar loading station and the loading time of the car, as the solar loading station is only relevant, if the electrical loading station is inactive. There is also a Default Event to indicate more possible influences on the loading time (i.e. defect in the car).

This representation shows concrete causality, as it presents relations between attributes, which change through events (weather gets sunny, engineer activates the electrical loading station). It is also possible to model a conditional relation, where we had to group multiple conditions in previous examples. Further it is the only example, which handles unknown information.

2.3.9 Multilevel Flow Model (MFM)

This representation of causality relations presented in paper [Lin11] is the MFM. It is used to present large systems with all entities and relations. The two basic concepts

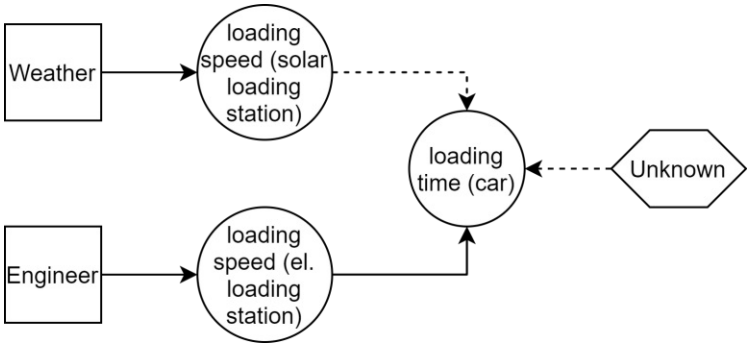


Figure 2.15: Dynamic Uncertain Causality Graph for the solar loading station example

of MFM are the means-end concept and the part-whole concept, where the means-end concept describes a concept, in which a system is modeled in a way, where in each step the system gets closer to a goal, while the part-whole concept describes a representation, where a system is split into multiple parts to present the whole system. In MFM a system is split up into goals and functional flow structures, which are interconnected to form a complete representation. There are different relation types to model different relations. Each relation type describes the influence of functional structure onto another one or a goal. Therefore, each relation can also be seen as a set of causality relations. In Figure 2.16 the basic representation types of MFM are shown.

In Paper[Lin11] a mill is used as example for MFM. The fitting diagram for the mill is shown in figure [example MFM mill]. We can see the system is split into four flow structures, where the first one (S3) is the water flow to the wheel and from the wheel out of the mill, the second (S2) shows the conversion of kinetic energy into rotational energy into heat and energy to energize the mechanical linkage to move the grinding stones inside the mill. Additionally, S1 is the grinding of the corn into flour and shells. These three structures are connected through mean-end relations as after each structure, we get closer to the goal of producing flour. The last structure (S4) is a lubrication system, which oils the wheel. It is not connected through a mean-end relation, as it is a precondition, but not a step in the process. The relation type applied on this example is a control relation (enable), as the lubrication system enables the wheel to fulfil his function.

The running example about the solar loading station can be modelled in four functional flow structures and a goal presented in Figure 2.18. The goal of the system is to load the battery of cars. The first structure (S1) represents the flow from the weather (So1), which is transported (T1) towards the solar loading station. At the loading station the energy is first converted into electricity (C1) and afterwards transported (T2) into the storage of the loading station (St1). Energy which overflows the storage of the loading station is sent (T3) into the power grid (Si1). The second structure (S2) represents the loading process of cars, where we assume there are endless cars (So2) which need energy. Successively each car drives next to the loading station (T4), where the car is

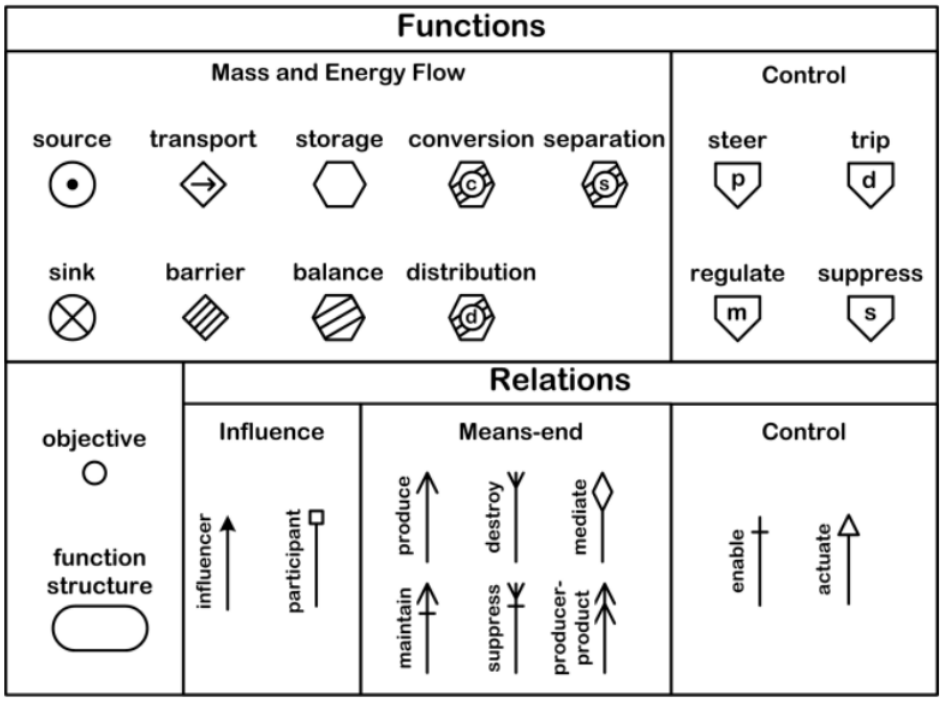


Figure 2.16: Structure elements of a Multilevel Flow Model, Reproduced from [Lin11]

loaded(D1). The distribution notation is used as the energy is distributed between all cars. Afterwards the loaded car leaves (T5) the loading station. The third structure represents the electrical loading station, which gets energy (T6) from the power grid (So3) and stores the energy in the loading station (St2). If the battery of the loading station is loaded, overflowing energy is fed back into the power grid (Si3). The last structure (S4) is the engineer which has to maintain the current state (active) of the electrical loading station (M1) to enable it to work. Between S1, S2 and S3, S2 we can see a producer-product relation, as the loading station provides the energy to load the cars. We further see a maintaining relation between S2 and O1 as the loading station has to keep providing power to cars to successfully load cars. At the end we see an enable-relation between S4 and S3 as the engineer has to maintain the current status to enable the electrical loading station to provide energy.

2.3.10 Event-Model-F (causality pattern)

The Event-Model-F, shown in [SFSS09], defines multiple patterns to represent events and relations between events. The causality pattern (shown in Figure 2.19) is described by two events, the effect and the cause. Further both, events are connected through a relation, which has a justification, which can be anything to additionally explain the causality relation. The paper lists an opinion, a theory or a scientific law as examples for possible justifications of a causality relation between these two events.

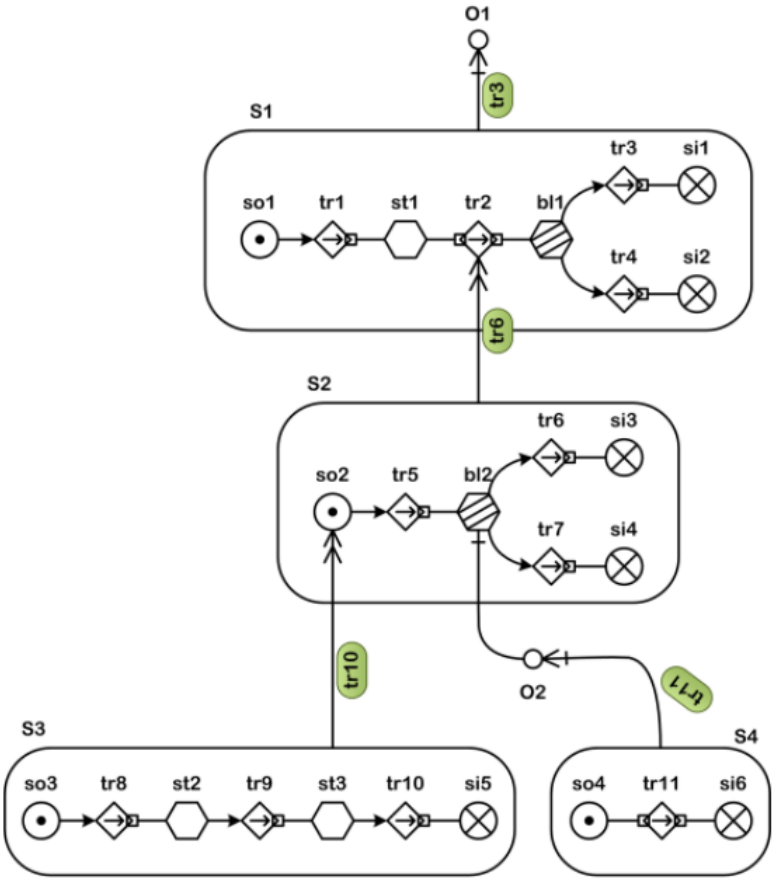


Figure 2.17: Example of a Multilevel Flow Model, Reproduced from [Lin11]

In Figure 2.20 the Event-Model-F causality relation is applied to our running example. Each event is either cause, effect or both. The type of the event is indicated at the start of the name. Further each relation has the required justification, which is modeled in orange. As this approach works with events, it can only model a concrete causality relation. Further, the electrical loading station (active/inactive) and the weather is combined into a single event, as this approach can not model conditional relations and therefore the exact description of the relation has to be modeled in the justification.

This formal approach for causality representation can present every possible relation due to the justification, which can be chosen independently of the relation. This enables or forces the user to create an own categorization of different types of relations and events. The user has to describe conditional relations in the justification.

Figure 2.18: Multilevel Flow Model of the solar loading station example

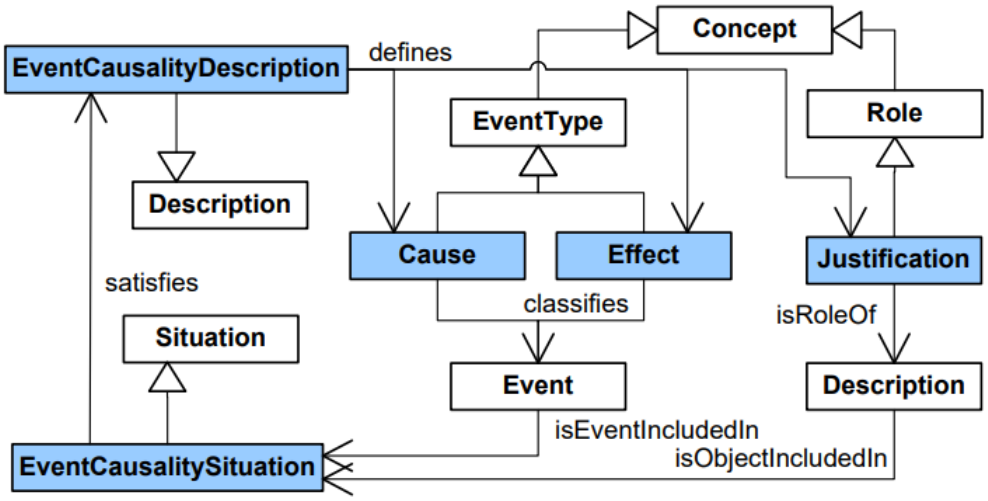


Figure 2.19: Abstract representation of the causality pattern of Event-Model-F, Reproduced from [SFSS09]

2.4 Summary

To summarize this section, we can see an overview of advantages and disadvantages of different representations in Table 2.5. In the following paragraphs we will discuss these dimensions and show their importance for our example.

Different Types of Causality Relations

It is important to model different types of causality relations and events, as there are different kinds of influences. Each relation type represents another important information. Applied to our example we can argue, it is important to know, if there is a different type of relation between the solar loading station and the car and the electrical loading station and the car, where the electrical loading station is always used if it is active. If we use the same type of relation, we will have to extend our effects and causes to contain this information (as we did in the example for the Fuzzy Cognitive Map). In Table 2.5 we can see, half of the approaches to model causality use different types of causality relations.

Influence Factor of Causality Relation

Another important factor is the influence factor of a causality relation, where each causality relation can be compared with other relations to show the strength of influence of one relation to another. In our example this can be used to provide thresholds or the correlation between two entities of a system. If we take the relation between sunny weather and a high loading speed, we can use the influence factor to show the point at which the loading speed decreases, when the weather gets worse or show the factor

Figure 2.20: Example of the causality pattern of Event-Model-F

in which the loading speed declines. The Influence Factor was only proposed by two approaches of the analyzed ones ([SFSS09], [Maz09]).

Representing a timeline

The NLP approaches propose a dimension to clarify the timeline, in which the events happen. Therefore, if the cause happens before, after or during the effect. This is important to consider for our example as it shows, if the sun has to shine before the loading process starts or if it is enough if the sun starts shining during the loading process to increase the loading speed.

Combination of multiple causes

This dimension enables an effect to have multiple causes. Therefore, each effect can have between 1 and n causes, where each cause on his own can cause the event or all causes together cause the effect. Applied to our example, we can use this to model the influence of the different solar loading station onto the car, where a high loading speed at the car can have different causes. Either the engineer activated the electrical loading station, or the weather is sunny and therefore the loading speed of the solar loading station is high. In the comparison of different approaches in the previous section, three approaches provide an explicit differentiation.

Generic vs. Concrete Causality

The last dimension describes, if the approach presents concrete or generic causality according to our definition. In the table we can see half of the approaches can be categorized as generic and the other half as concrete ones.

Result

At the end we can conclude to build an appropriate representation of causality in a system, which strongly depends on the use case to choose a correct representation. For our example different approaches are viable. We could use the Event-Model-F as it is possible to describe each relation with the additional justification, to provide every information needed. Alternatively we can use the DUCG approach to model the conditional relation between the solar loading station, the electrical loading station and the car, where the solar loading station is only used, if the electrical loading station is not active. Additionally, it is also possible to provide a clear definition of root causes with the help of DUCG. In the end it is also possible to model the example with the help of an SDG to provide a more generic overview of the system. The SDG is a simple representation, which provides each relation in a single graph.

Model	Different Types of Causality Relation	Influence Factor of causality relations	Represents a timeline	Combination of multiple causes	generic/ concrete causality
Extended SOSA approach	no	no	no	no	generic
Event-Model-F	yes	yes	no	no	concrete
Adjacency & reachability matrix	no	no	no	no	generic
Signed Directed Graph	yes (positive/negative)	no	no	no	generic
Matrix Layout Plots	no	no	no	no	generic
Fuzzy Cognitive Maps	no	yes	no	no	concrete
Influence Diagram in Canonical Form	no	no	no	yes	concrete
DUCG	yes	no	no	yes	concrete
MWM	yes	no	no	yes	concrete
Nature Language Approaches	yes	no	yes	no	generic

Table 2.5: Comparison of different causality relation representations

CHAPTER 3

Application of causality models in CPS

In this chapter we aim to investigate RQ2, where we focus on scenarios of causality acquisition and causality representation in already existing systems. In Section 3.1 we briefly describe these systems and point out interesting aspects of each of those systems. Section 3.2 concludes this chapter with a comparison of these systems and proposes a representation, considering the aspects of Section 2.4 and Section 3.2, which can be used in the BIFROST simulation engine. The comparison of the systems and the proposed representation, presented in Section 3.2 are contributions of the thesis.

3.1 Approaches of Causality Application

This section shows approaches to acquire and represent causality. These approaches aim to acquire causality from a Cyber Physical System, present it in a formal way and evaluate it to find root causes and optimize different variables in the system.

3.1.1 Semantic Smart Building Diagnoser

The Semantic Smart Building Diagnoser, presented in [PSL14], is applied to buildings to evaluate possible causes for anomalies appearing during operations. To represent causality, a semantic model is used, where the model is an extension of the Semantic Sensor Network skeleton ontology defined by the W3C incubator group. The model uses different classes to represent a feature of interest, which is an observed area (i.e.: room), an actuator, a property of an actuator, (un)observed values and anomalies. Additionally, it represents relations between these classes, where a relation can be either a property linked to the actuator, an (un)observed value linked to a feature of interest or a cause-effect-relation, which can be positive or negative, between either of these classes.

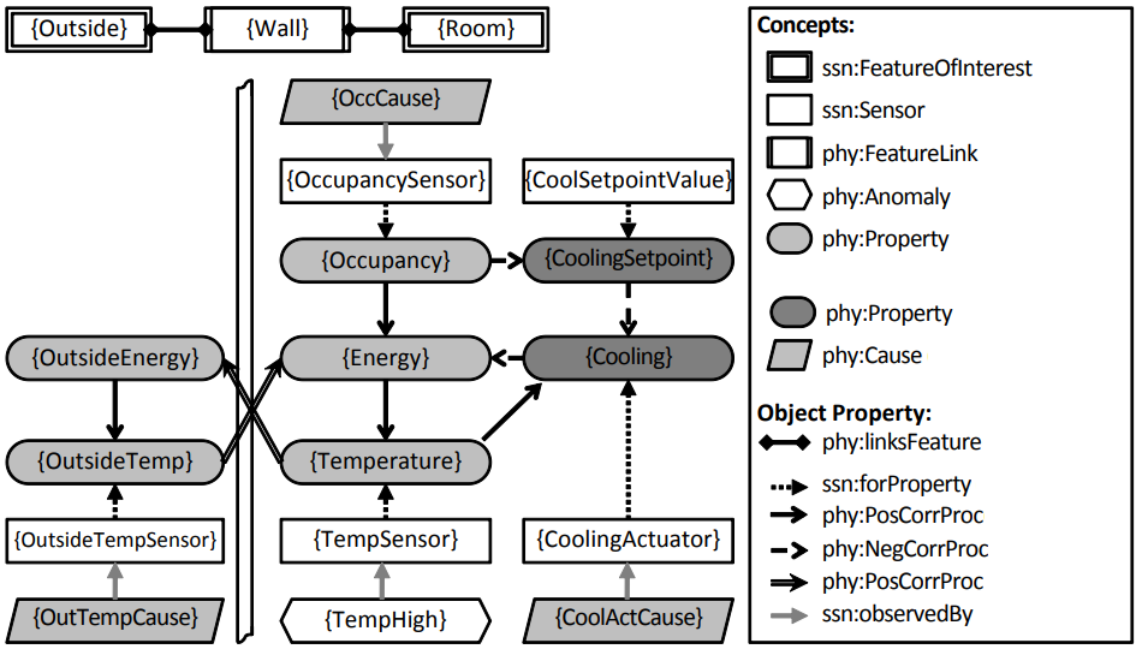


Figure 3.1: Example of the Semantic Smart Building Diagnoser, Reproduced from [PSL14]

To build a semantic representation for the Semantic Smart Building Diagnoser, each object in the building has to be labeled according to possible types, which are different for each building. The paper presents a semi-automated system to annotate automatically each object. These annotations must be manually corrected by an expert. Afterwards, each feature is related to each (un)observed value and each object to each property of the object. Further, the objects are interconnected with values they observe. Therefore each object is linked with properties and observable values. Unobservable values are still represented in the model but not connected to their influences.

With this model it is feasible to evaluate the cause of anomalies. To find the cause, it is possible to find a sequence of relations between properties and observed values to find the root cause. In the paper an example is presented, where a room has an occupancy sensor, a cooling setpoint, a temperature sensor and a cooling actuator. Additionally, the temperature outside the room is modelled. The objects, properties and observed values of the example can be seen in Figure 3.1. If the anomaly “high temperature” appears, there will be multiple sequences of relations and therefore, multiple possible causes. First of all, a high temperature can be caused by a high occupancy, as it is linked through the observed value “energy”. Alternatively, it can be caused by high temperature outside of the building as these are linked through “temperature”, “energy” and “outside energy”. The last possibility is the cooling actuator, as they are linked through “temperature” and “cooling”.

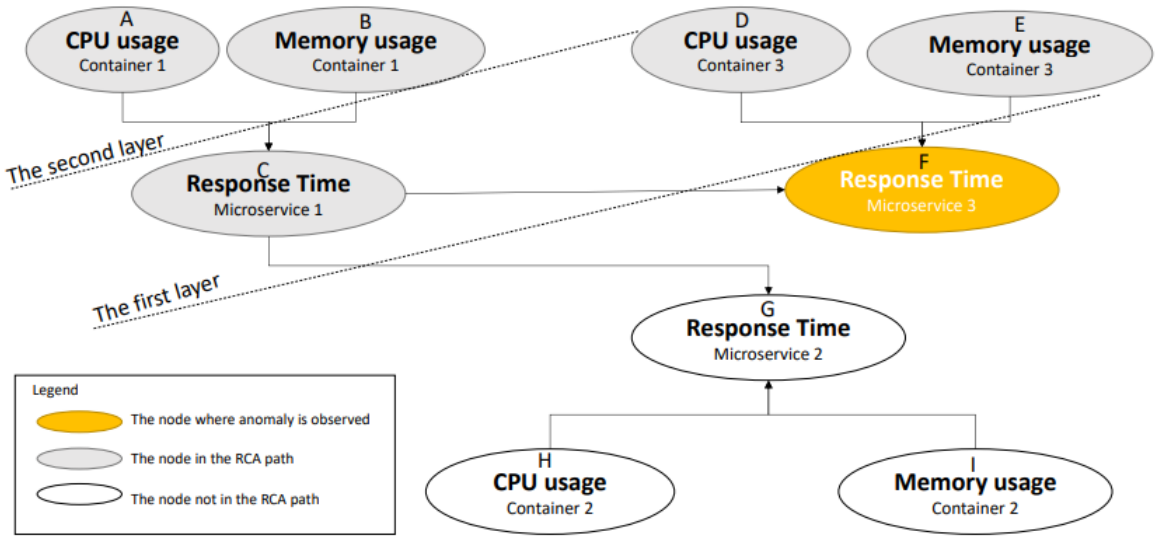


Figure 3.2: Example of a O&M Knowledge Graph, Reproduced from [QDY+20]

3.1.2 O&M Knowledge Graph

In this application of causality representation, shown in [QDY+20], a weighted directed acyclic graph is used to find the cause of an alarm in a cloud-application. To construct the graph, the paper proposes an algorithm, where a complete graph is built in the first. Each node represents a performance indicator. Afterwards the algorithm decides, if there is a causal relation between each pair of performance indicators. In this cause, the edge between these two performance indicators persists, otherwise it is deleted. The result of the first step is a graph, containing undirected edges. Afterwards the algorithm uses statistical methods and interference rules to direct and weight each remaining relation.

It is easy to use this graph, to find a root cause, as there are only two rules, which can be followed by a ranked list of possible root causes. The first rule is, the higher the weight, the higher the priority. The other rule is the shorter the path, the higher the priority. In the given example these rules are applied to the graph seen in Figure 3.2, where an abnormal response time in container 3 is observed. In the graph we can see there are four possible root causes for the anomaly. The most likely outcome is the memory usage in container 3 as it is weighted as 0.7 and only has a pathlength of one. Compared to the Memory usage of container 1, where the weight is 0.7 as well, but the path length is two. These two possibilities and the other two can be seen in Figure 3.3.

3.1.3 Root Cause Map

In Paper [RH04] a more general approach to collect causality information is described. It uses a root cause map to model effects with their possible causes. The whole map shows each cause related to each effect, with all steps which have to appear in between for the

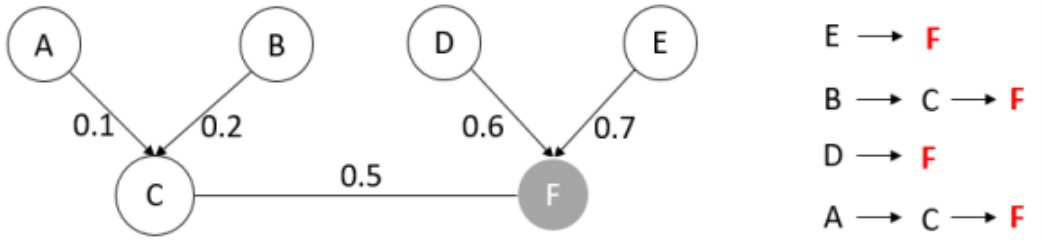


Figure 3.3: Evaluation of a O&M Knowledge Graph, Reproduced from [QDY+20]

cause to happen. The building of a root cause map involves four steps. In the first step the data has to be collected, therefore the object, which has to be analyzed, has to be measured and data has to be saved in a logic way. Afterwards the causality relations have to be extracted from the data. The paper proposes a sequence diagram to visualize all possible cause sequences to trigger an effect. Afterwards, these diagrams must be used to identify all possible root causes for each possible effect. This representation is called the root cause map. From this root cause map we can extract possible recommendations to avoid or to trigger a cause, which in the end is used to build a root cause summary table, where each row contains the causal relations, which lead to an effect.

3.1.4 GoalArt

Goal Art, presented in [LÖC07], is an algorithm which uses a Multilevel Flow Model (presented in Section 2.3.9) to extract and model causality. GoalArt is used for the root cause diagnosis in a power grid. Therefore, each element in the Multilevel Flow Model presents an object in the grid. Furthermore, each object is assigned to one of the following four types: generator, line element, bus bar or load element. A transformer for example is a line element, as it is a power transporting device. The algorithm can assign the correct type for each of the objects automatically. As each functionality of the power grid is modeled as function in the MFM and each goal is a possible effect, it is easy to track back possible causes, because each function of an MFM is interconnected with others to model all cause-effect relations of the system.

3.1.5 Produced Water Treatment

The simulated water treatment facility, introduced in Paper [NJZ+18], also uses a MFM to present possible causes. As they work in a simulated environment, they have defined certain threshold limits for alarms. Compared to GoalArt, this approach has the MFM modelled by an expert. Therefore, the system has no automation to model an MFM. This approach uses each goal as a possible alarm state and each functionality of the system as a function in the model. With the help of this construction it is possible to find the correct root cause by tracking back the functionality, starting at the goal modelling the alarm, currently active.

3.1.6 WARP

WARP, presented in [MDEG18], is an engine developed to preprocess streams of events. Therefore, it receives all events from the observed engine, groups these events and filters them if needed. Therefore, it is possible to show a single event for a single anomaly instead of multiple events of a single cause, because each element between cause and last effect can trigger an alarm. WARP offers the possibility to filter events in the following ways. First it is possible to filter due to semantics, where events with a different type of data or based on interference rules are chosen. Furthermore, it is possible to filter topologically, where events from the same area of a grid or events with the same source of data are chosen. It is also possible to filter geographical data, where WARP uses GPS or similar technologies to determine other possibly related events. As a last filter, it is possible to filter events based on a heuristic, where previously learned data can be applied to find related events.

WARP is not able to provide an automatic evaluation of causes, but it can present a structured overview over every event that has happened to an expert, who then has to decide further, which further actions can be taken to avoid future alarms.

3.1.7 Framework for a smart data analytics platform towards process monitoring and alarm management

This framework, shown in [HSC18], is used to present causality relations between different alarms. Therefore, it is possible to find a root alarm, based on the causal relations. To represent these relations, the framework uses Availability- and Reachability matrices. Further, the framework is able to model a Signed Directed Graph from the initial matrix. To build these representations, the framework uses a statistical method, called Transfer Entropy (in detail explained in [DYCS13]). This method can calculate the causal strength between different alarms. In Paper [HSC18] there is an example based on the SDG, which can be seen in Figure 3.4. In the first step each causal relation is modelled in the graph (left picture). Afterwards, the framework is using a normalized value of the causal strength to find indirect causal relations and removes them. The result can be seen in the right picture of Figure 3.4.

3.1.8 Building energy managment system

In this last presented approach to acquire and represent causality, a set of interference rules is applied to optimize a building automation system. These rules are created by an expert or extracted by data mining previously collected data. The whole energy management system has different steps, until the optimization can be applied to the building. This process is shown in Figure 3.5.

At the start of the process, the system receives current information about the building and the weather. Afterwards the workflow splits into two different modules to evaluate optimization potential. In one module the received data is transformed into a structured,

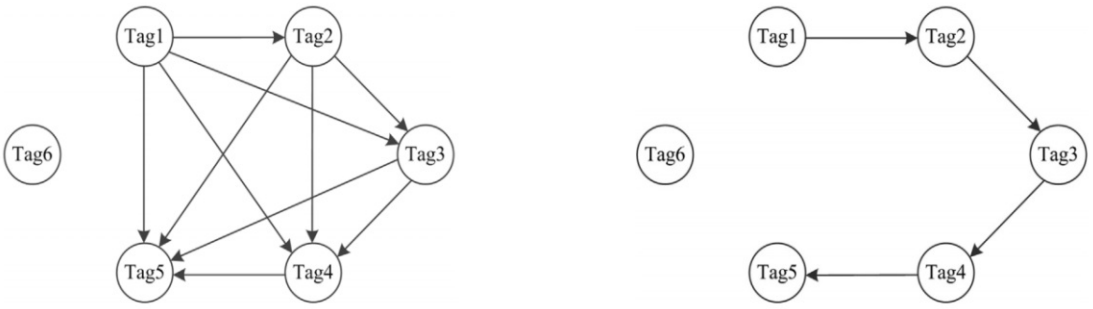


Figure 3.4: Example of a SDG for process monitoring and alarm management, Reproduced from [HSC18]

formal representation, while the other module calculates thresholds for certain values and expected values for parameters. With the help of both calculations, the system is able to find inefficiencies in the building. For each parameter that is marked as efficient, the system does not need to take further actions. Every other parameter is brought into context with surrounding objects and parameters and the system decides, if the parameter is abnormal or not. For example, if the temperature is high, the window is open and it is hot outside, it can be followed that the high temperature is intended. On the other side, if the window is closed and a heater is active, there is an abnormal behavior. If the system detects abnormal behavior, it will find the cause through the set of interference rules and sends a recommendation on how to solve the abnormality to the building system, which then can decide, if it wants to apply the recommendation or not.

3.2 Summary

At the end of this section we summarize different aspects of the application of causality extraction and modeling in simulated and real-world scenarios. In Table 3.1 a comparison of these applications can be seen.

Application Area The application area is quite important for different causality extractions, as it highlights the different needs in the model for different areas of application. For example, in Paper [PSL14] a smart building is presented, where it is important to explain abnormal high or low temperature. Therefore they chose a skeleton ontology to solve the problem. On the other side, in [LÖC07] a smart grid has to be modelled, where a Multilevel Flow Model is a proper solution.

Expert knowledge required to build the model. This dimension shows, if expert knowledge is required to build the initial model. Therefore, it shows the required effort to build a model. In some of the presented approaches it is necessary to use an expert to build the initial information model. In [LCH+19] the used interference rules have to be

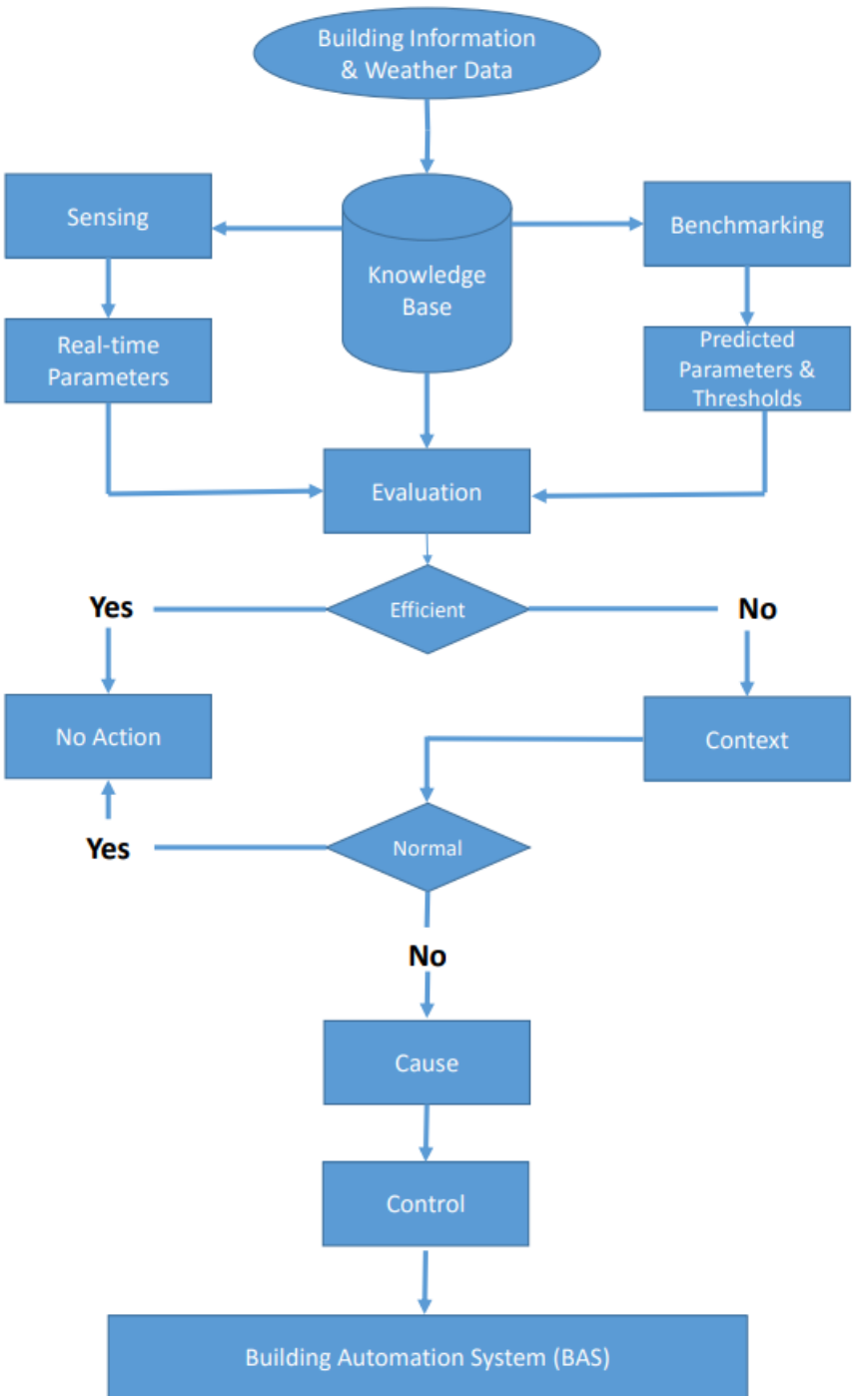


Figure 3.5: Workflow of the Building energy management system, Reproduced from [LCH+19]

set manually, in addition to the found rules, based on old data. Further the approach [PSL14] can only semi automatically build the model and needs an expert to double check the automatically generated model. On the other hand, the algorithm, introduced in [LÖC07], does not need an expert to build the MFM, as the algorithm can do it on his own. Half of the presented approaches can build a model on their own.

Expert knowledge required to run the model The last dimension described is the necessity of expert knowledge to run the system, after the initial creation of the model. Therefore, if there has to be an expert, which doublechecks causes, generated by the system or decides based on a list of causes provided by the model, the root cause. An example for a system, requiring an expert is WARP, shown in [MDEG18] , where the system only provides an overview over each appearing event. In contrast to WARP, [PSL14] can provide an automatic explanation for each occurring anomaly. Therefore, the Semantic Smart Building Diagnoser does not need an expert during operation. The only other system which does not need an expert is the Building Energy Managment System, presented in [LCH+19]. no

Result At the end we can summarize, that depending on the application area, there are multiple different approaches for the extraction and modelling of causality. Overall, there is one representation which appears to be superior compared to others, the Multilevel Flow Model. This representation is not only able to show causality without a lot of statistical calculations, but can also provide an overview over large systems (e.g.: smart grid) and even find anomalies based on the modelled functionality of the systems. Further it is possible to extend the Multilevel Flow Model with the different aspects described in Section 2.4.

Approach	**Application area**	**Expert knowledge required to build the model**	**Expert knowledge required to run the model**
Semantic Smart Building Diagnoser	smart building	yes	no
O&M Knowledge Graph	cloud computing	no	yes
Root Cause Map	general root cause analysis	yes	yes
Goal Art	smart grid	no	yes
Produced Water Treatment	root cause analysis in PWT	yes	yes (can be automated over time)
WARP	distributed grid	no	yes
Framework for a smart data analytics platform towards process monitoring and alarm management	root cause analysis in a large system	no	yes
Building energy management systems	smart buildings	yes	no

Table 3.1: Comparison of different approaches of Causality Applications

CHAPTER 4

Causality Acquisition

This chapter focuses on RQ3 and investigates different possibilities to acquire causality relations from a CSP. In Section 4.1 we formally define the problem and therefore extend the definition of causality, originally presented in Section 2.1. This problem definition is the first contribution of this chapter. Based on the problem definition of Section 4.1, we present an overview of five different methods to acquire causality in Section 4.2. Finally, Section 4.3 concludes the chapter by presenting an overview of important aspects which have to be considered while using one of the causality acquisition algorithms presented in Section 4.2. Further, the overview presented in Section 4.3 discusses the applicability of these causality acquisition algorithms to the BIFROST simulation engine. The overview of Section 4.3 is another contribution of thesis.

4.1 Causality Acquisition Problem Definition

For the purpose of this thesis we use the SOSA approach, presented in Section 2.3.7, to define causality acquisition. Especially we introduce the feature of interest, the property and the causal relation. A feature of interest is an object in the system. Each object has multiple properties. A feature of interest can be a house with the properties “energy level” or “power level”. Each feature of interest can have multiple properties, but each property belongs to a single feature of interest. The presented causality acquisition algorithms aim to collect causal relations between these properties, where a causal relation provides a property (cause), which causes another property (effect).

In this thesis the aim is to evaluate causality acquisition algorithms in the context of the BIFROST simulation engine. BIFROST provides settlements, which can contain different types of buildings and a different number of each of these buildings. Each building is represented as a feature of interest. Further BIFROST provides different types of measurements from each building, which are the properties of the feature of interest. Each type of building collects the same types of properties. These property

measurements are collected by an expCPS, presented in [AES+20]. This collection of measurements can be transformed into a time series by the framework, presented in Chapter 5. Therefore, we define the following terms:

- **Settlement**: A settlement S is a tuple which describes the whole CPS. Therefore, each settlement contains:
  - a set of Features of Interest FI
  - a set of Feature Types FT
  - a set of Properties P
  - a set of Property Types PT
  - a set of timeseries TS
  - the mapping between a Feature of Interest and a Feature Type, represented by the function $FIFT : FI- > FT$
  - the mapping between a Property and a Property Type, represented by the function $PPT : P- > PT$
  - the mapping between a Feature of Interest and a Property, represented by the function $FIP : FI- > P$

  Formally a Settlement is represented in Formula 4.1. The different parts of the tuple are described in the following points.

$$S = \{FI, FT, P, PT, TS, FIFT, PPT, FIP\} \tag{4.1}$$

- **Feature of Interest**: A Feature of Interest fi of FI is a single object in the observed system. For an object to be classified as Feature of Interest it has to contain relevant information which researchers want to collect. For example, a Feature of Interest in the BIFROST simulation engine can be a building or a transformer.
- **Feature Type**: To compare different Features of Interest, each Feature of Interest belongs to a Feature Type ft of FT. Each Feature of Interest has a single Feature Type, while multiple Features of Interest can belong to the same Feature Type. Therefore, for each Feature of Interest there exists a mapping function $FIFT(fi) = ft$, which maps the Feature of Interest to the corresponding Feature Type. An example for a Feature Type is *police_station* and $fire\_station$, where each Feature of Interest is either a *police_station* or a $fire\_station$, but there can be multiple *police_stations* in the same settlement.
- **Property** A property p of P is a single measurable value of a Feature of Interest. Each Feature of Interest can have multiple properties, while each property belongs to a single Feature of Interest. For example, a building (Feature of Interest) has

the property *energy_level* and *power_level*. As a property belongs to a Feature of Interest, we can map a property to the Feature of Interest with the following function: $FIP(p) = fi$.

- **Property Type** Similarly to the Feature Type, we also categorize Properties to enable a comparison. Therefore, we introduce the Property Type pt of PT, where each Property has a single Property Type, while multiple Properties can belong to the same Property Type. If two different Properties p_1 and p_2 of P are given and the set of Property Types $PT = \{energy\_level, loading\_level\}$, p_1 and p_2 can belong to the Property Type *energy_level*, but neither p_1 nor p_2 can belong to both *energy_level* and *loading_level*. Therefore, a mapping function $PPT(p) = pt$ exists for each Property.
- **Datapoint**: A Datapoint d is a single measurement taken from a property and is represented as a tuple of the time when the measurement was taken and the value of the measurement. Therefore a Datapoint $D = (v, t)$, where v is the measured value and t the time of the measurement.
- **Timeseries**: A Timeseries ts of TS is an ordered set of Datapoints and therefore a set of multiple measurements of the same property. Therefore, each Timeseries belongs to a single property. Formally a Timeseries is represented as $ts = \{d_1 \dots d_n\}$, where n is the number of Datapoints in the Timeseries and ts is in TS. Further a single Datapoint is represented by $ts(t)$, where t is the time of the Datapoint we want to represent.
- **Causality Relation/Causal Relation** The causality relation represents a causal relation between two Properties. Therefore, formally a causal relation is represented as $c(p_1, p_2)$ where p_1 and p_2 are in P and belong to a Feature of Interest in the same Settlement. Further p_1 is the cause while p_2 is the effect of the relation. An example for a causal relation is $c(loading\_level, active\_power)$, where a higher *loading_level* (Property) of a $transformer$ (Feature of Interest) causes a higher *active_power* (Property) of *residential_house* (Feature of Interest).

With this terms we focus on solving two problems, which aim to solve RQ3. These problems are defined as follows:

- P1: Given two properties p_1 and p_2 of P. How is it possible to determine if c(p_1,p_2) holds true? Therefore, the output of problem P1 is true, if c(p_1,p_2) exists and false, if c(p_1,p_2) does not exist.

  For the setup in BIFROST, we always compare two timeseries ts_1 and ts_2 of two Properties p_1 and p_2, where ts_1 is a timeseries of p_1 and ts_2 is a timeseries of p_2. Therefore a causal relation can also be shown as: $c(ts_1, ts_2)$.
- P2: Given a Settlement S, how is it possible to get a result set R_S of Causal Relations, where each Causal Relation in R_S represents an existing Causal Relation

between two properties in the Settlement S. Formally $R_S = \{c_1, ..., c_n\}$, where n is the number of causal relations in Settlement S.

Therefore, this chapter aims to provide an overview of different causality acquisition algorithms that could address the problem as defined above and a discussion on the possibility to collect these causal relations in the BIFROST simulation engine.

4.2 Causality Acquisition Algorithms

The presented Causality Acquisition Algorithms (CAA) aims to solve P1, where two timeseries ts_1 of p_1 and ts_2 of p_2 are the input of each CAA. After applying the algorithm, each CAA provides the information if $c(p_1, p_2)$ exist. The output of each CAA is therefore true or false, which is the output required by P1.

To solve P2, each CAA is applied to each possible combination of properties available in a Settlement S. Even if an expert would recognize combinations of properties, which can logically have no causal relation, the algorithm is not able to distinguish this type of situations. After applying each CAA to each combination of properties the algorithm collects the positive results into a result set R_s, which contains each causal relation c of S. The different CAAs to build R_s are detailed in the following Subsections.

4.2.1 Granger Causality

Granger Causality is developed by Granger [Gra69] and designed to compare two timeseries x and y in order to find a possible causal relation. To correctly predict causal relations, as presented in [CSK+18], between timeseries x and y of TS, it compares the possibility to predict $x(t)$ with $x(t-l)$ and $y(t-l)$ and $x(t-l)$ alone. Based on this comparison it can be followed, that the prediction is increased, there is a causal relation, otherwise not. The variable l in this calculation stands for the time lag between the value $x(t)$ and the value of the past $x(t-l)$, with whom $x(t)$ is compared.

Formally the Granger Causality between two variables is given by a bivariant autoregressive model, which can be expressed as shown in Formula 4.2 and 4.3, where the variables A-D express the regression coefficient matrix. This matrix is built with the help of the ordinal least square method described in [CI11]. ϵ_t and η_t are white noise processes, which simulate the error function in the calculation. To finally compare these two models, an F-Test is used, which compares the variance of the two outcomes.

$$x(t) = \sum_{\tau=1}^{m} A_\tau \cdot x(t-\tau) + \sum_{\tau=1}^{m} B_\tau \cdot y(t-\tau) + \epsilon_t \tag{4.2}$$

$$y(t) = \sum_{\tau=1}^{m} C_\tau \cdot x(t-\tau) + \sum_{\tau=1}^{m} D_\tau \cdot y(t-\tau) + \eta_t \tag{4.3}$$

Granger causality can be applied to the BIFROST simulation engine, but it is possible for it not to detect each causal relation as Granger Causality works more accurate on linear functions. The other requirements for the application in BIFROST are fulfilled, as it is possible to extract timeseries from BIFROST.

4.2.2 Transfer Entropy

The Transfer Entropy, first introduced by Schreiber [Sch00], is a method to measure the information flow between two time series. Therefore, it is possible to measure the information exchange between two information flows in each direction on its own. The Transfer Entropy is based on the Shannon Entropy, shown in Formula 4.4, which describes the uncertainty of an information. This uncertainty is given as H_I in the formula. An example for Shannon Entropy is a coin flip, where the most uncertainty is given, if the coin is unbiased as we cannot expect either of both possible outcomes. In contrast, a biased coin, which always has the same outcome, has no uncertainty as we can know the outcome before the coin is flipped. Based on this concept, the Transfer Entropy is extended to two different information sources which measure the information with a time delay. The formula for the Transfer Entropy is given in Formula 4.5, where we see the extension for the calculation of uncertainty for a single event towards the measure of information flow between two sources of information. Further explained Transfer Entropy measures the reduction of uncertainty for a future value of the first timeseries, based on the knowledge of an old value of the second timeseries. In the formula i and j are the two timeseries, while i_n and j_n represent the values at a certain datapoint.

$$H_I = -\sum_i p(i) log_2 p(i) \tag{4.4}$$

$$T_{J\to I} = \sum p(i_{n+1}, i_n^{(k)}, j_n^{(l)}) log \frac{p(i_{n+1}|i_n^{(k)}, j_n^{(l)})}{p(i_{n+1}|i_n^{(k)})} \tag{4.5}$$

The result of the calculation is still based on the chosen values, where we can set two different parameters. The first one is the historical length (k, l), which decides, how many previously measured values are relevant for the next current one. For this parameter Schreiber proposes either $k = l$ or $k = 1$ to reduce computational effort. Furthermore, it is possible to adjust the delay between the compared values of the cause and the effect. Therefore, we can adjust the timespan between the cause happening and the effect appearing.

There are different estimation methods, which are used by the Transfer Entropy method to implement the probability measure $p(i_{n+1}, i_n^{(k)}, j_n^{(l)})$. In the following, different methods to calculate the probability measure are presented.

Kernel Density Estimator

A Kernel Estimator is a function, presented in [Liz14], which is used to generate a probability distribution function (PDF). These functions show the distribution of a single specified value, similar to histograms. Compared to histograms, the Kernel Estimator is represented as a curve instead of a collection of bars. To create a Kernel Density Estimator, multiple Kernel Estimators are combined into a single curve. Applied to the Transfer Entropy, this means we create a Kernel Estimator for each value of the time series and combine these estimators to build the probability measure.

To create a smooth curve, the kernel function uses a bandwidth to round off the result. The result of a Kernel Estimator is heavily biased towards the chosen bandwidth as the result is quite different, as seen in Figure 4.1. Figure 4.1 shows seven different Kernel Estimators (in color) and the Kernel Density Function (in grey). Further, different bandwidth values (h) are used in each diagram to show the bias of the resulting function towards the chosen bandwidth. The higher the bandwidth gets, the more biased the function gets. The lower the bandwidth gets, the higher is the risk of the function to not provide useful information.

Formally the probability measure, created with the Kernel Density Estimator, for two timeseries is given in Formula 4.6, where N is the number of properties, while Θ represents the kernel functions, which defaults in the used implementation to a step kernel, which is given in Formula 4.7 and 4.8. Finally the bandwidth is presented as parameter r.

$$\hat{p_r}(x_n, y_n) = \frac{1}{N} \sum_{n'=1}^{N} \Theta(\left| \begin{matrix} x_n - x_{n'} \\ y_n - y_{n'} \end{matrix} \right| - r) \tag{4.6}$$

$$\Theta(x > 0) = 0 \tag{4.7}$$

$$\Theta(x \leq 0) = 1 \tag{4.8}$$

Kraskov Estimation

Similar to the Kernel Density Estimation, described in [Liz14], the Kraskov Estimation also uses multiple kernel functions to provide a probability measure. The difference between these two approaches is the calculation of the bandwidth. While the Kernel estimator takes a fixed value for the bandwidth, the Kraskov Estimation calculates the bandwidth based on the k nearest neighbors algorithm. In this approach to build a kernel function, the algorithm takes the k nearest neighbors next to the calculated value and defines the bandwidth as the average of the distance between these samples.

For this estimation method it is not necessary to set a fixed bandwidth and therefore the bias created by the bandwidth choice is prevented. On the other side it has a significant higher calculation time, as the bandwidth has to be calculated for each kernel density function.

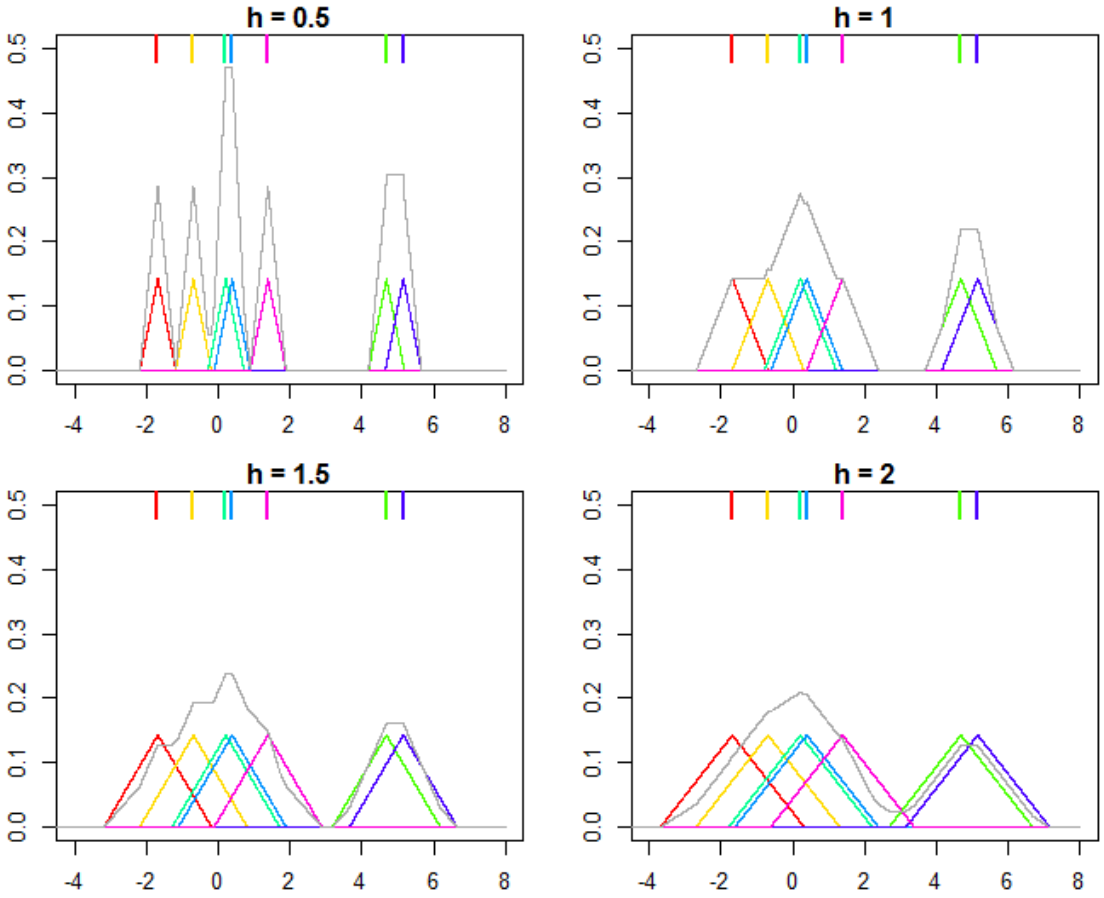


Figure 4.1: Example of a Kernel Density Graph with different bandwidths, Reproduced from [23]

Applicability for BIFROST simulation engine

The Transfer Entropy can be applied to the BIFROST simulation engine, as it is possible to extract timeseries for each property measured in the simulation. After the extraction it is further possible to compare the timeseries, based on a Transfer Entropy implementation, to find a possible causal relation.

4.2.3 Peter-Clark Algorithm

For this causality acquisition algorithm, proposed by [SGSH00], an algorithm is used, which starts with a graph G, where each vertex, which represents a property of a feature of interest, is connected with each other. The Peter-Clark Algorithm is not able to solve P1, but solves P2 directly. In the first step, the algorithm tests each edge for conditional independence based on a set of neighbor vertices, which increases each test round, starting with the empty set. The result of the first step is an undirected graph with all appearing

causality relations. In the second and last step the algorithm directs each undirected edge to build a causality graph.

In Figure 4.2, an example for a Peter-Clark (PC) Algorithm, presented in [LHL+16] is shown. In this example the function I represents a conditional independence test between two variables. In the implementation, explained in Chapter 5, a Fisher Z test is used, but other conditional independence tests can be used as well. In the first test round, the conditional independence between each combination of values is tested and the edge between B and C, D and the edge between C and D are removed, as the algorithm has found conditional independence (based on the empty set as condition). In the second test round, the edge between A and B is removed as the algorithm finds A and B to be independent of each other if C is known. The remaining edges are tested as well, but no further edge can be removed, therefore the undirected graph is constructed. In the last step each remaining edge has to be directed to provide the final graph.

The Peter-Clark Algorithm has some major disadvantages, as the result depends on the order of the conditional independence tests. Therefore, different edges are removed, based on the order of calculation. Furthermore, the Peter-Clark Algorithm scales worst case exponential [LHL+16]. We therefore do not recommend it for the BIFROST simulation engine, as many timeseries and properties are expected, and the calculation time can be too high. Further, the result required by the expCPS should be consistent and not dependent on the chosen algorithm.

4.2.4 Neyman Rubin Causality

The Neyman-Rubin causal model, originally developed by Neyman [SNDS90], measures the outcome of an event under the assumption, that another event happens, or the other event does not happen. Based on these two outcomes (either the second event happened or not), the model generates two comparable outcomes. In the following step the algorithm compares the outcomes and decides, if they are equal or not. If the two outcomes are equal, the methodology can conclude there is no causal relation between the events as the outcome does not change, independent if the second event occurs or not. On the other hand, if the two outcomes are not equal, the model concludes, that there is a causal relation.

In the BIFROST simulation engine each event would be mapped to the current value of a property. Therefore, each property has to be measured with another property present and the other property not present. This is not possible in the BIFROST simulation engine, as only a whole Feature of Interest can be placed or removed. Further it is not possible to collect the two outcomes, independent of other influences. Therefore, we do not recommend the usage of the Neyman-Rubin causal model for the BIFROST simulation engine.

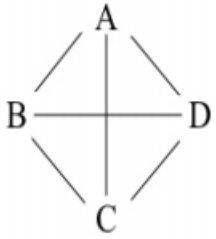
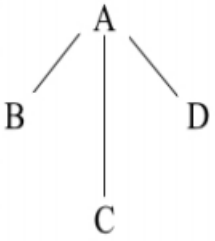
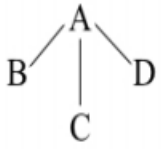
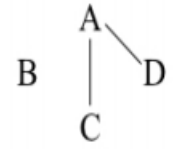
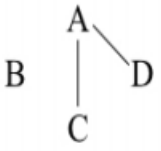
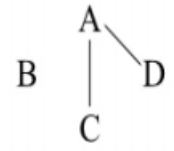
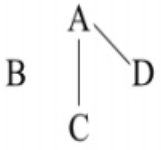
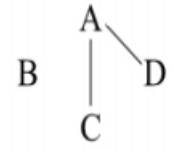
Level	Graph	#CI tests	Test	Result	Updated Graph
1		1	I(A, B)?	No	
		2	I(A, C)?	No	
		3	I(A, D)?	No	
		4	I(B, A)?	No	
		5	I(B, C)?	Yes	
		6	I(B, D)?	Yes	
		7	I(C, A)?	No	
		8	I(C, D)?	Yes	
		9	I(D, A)?	No	
2		10	I(A, B | C)?	Yes	
		11	I(A, C | D)?	No	
		12	I(A, D | C)?	No	

Figure 4.2: Example of a Peter-Clark Algorithm, Reproduced from [LHL+16]

4.2.5 Structural Equation Method

The main purpose of the Structural Equation Method, presented in [YDSC14] and [CSK+18], is to predict unobservable variables of a system. To predict these variables, the algorithm has to find causal relations beforehand and can therefore try to predict these relations.

To apply the Structural Equation Method to a system, the algorithm needs two different sets of variables as input: observable variables (properties) and unobservable variables. Further, the algorithm needs a prebuilt set of causality relations between the unobservable variables. After receiving the necessary information, the algorithm applies series of statistical methods to acquire further causal relations and predict values of unobservable

variables.

As the Structural Equation Method needs a set of prebuilt relations, it can currently not be applied to the BIFROST simulation engine, as these set of relation is not known at the current point of research. It is further not an ideal fit, as the main purpose of the algorithm is to predict unknown variables, instead of finding causal relations. Therefore, we do not recommend the usage of this method for the BIFROST simulation engine.

4.3 Summary

All presented methods use different algorithms and statistical methods to acquire causality information from data, but each of these methods works with different data and therefore requires different input data. In Table 4.1 a comparison of the different algorithms is shown.

The comparison is based on the input and output data, the possibility to apply previous knowledge and the applicability to the BIFROST simulation engine. The input and output data are important, as the required data changes the possibility to apply an algorithm to the collected data. Further it is important as the output has to be used by the system applying the causality acquisition algorithm. The possibility to apply previous knowledge to the current system is important, as the algorithm does not need to acquire this knowledge and can therefore increase the performance and reduce the calculation time. This can also enable an algorithm to be applied as enough previous knowledge can reduce the runtime to an applicable time. The last dimension is the applicability to BIFROST, which is the most important point of the chapter, as we want to apply the best algorithms to the BIFROST simulation engine and therefore it is important to know, if the algorithm can be applied to the BIFROST simulation engine.

4.3.1 Input data type

Based on the chosen algorithm, the type of expected input value changes. The most common input data type is a list of different timeseries, where each timeseries presents the change of a value of a feature of interest over time. Three of the five presented algorithms work based on these timeseries. Another possible input data type is the current value of an feature of interest in different scenarios. This input data type is used by the Neyman Rubin algorithm. The last occurring input data type contains timeseries and values, which are aimed to predict possible causal realtions based on the previous knowledge. Additionally, this input data type contains previous knowledge. This input data type is used by the Structural Equation Method.

4.3.2 Output data type

As the input data type varies based on the algorithm, so does the output data type. Three of the presented algorithms simply show, if there is a causal relation between the provided cause and possible effect. The other two algorithms either return a directed

Acquisition Algorithm	**Input data type**	**Output data type**	**Can handle previous knowledge**	**Can be applied to BIFROST**
Granger Causality	Timeseries	Causal Relations	no	yes
Transfer Entropy	Timeseries	Causal Relations	no	yes
Peter-Clark Algorithm	Timeseries	DAG	yes	yes
Neyman-Rubin Causality	values in different Scenarios for the same property	Causal Relations	no	no
Structural Equation Method	Observable and unobservable variables, relations between these variables	predicted values for unobservable variables	yes	no

Table 4.1: Comparison of different causality acquisition algorithms

acyclic graph to provide this information or only use the causal relations during the algorithm to provide predicted values of unknown variables.

4.3.3 Can handle previous knowledge

This dimension describes, if the algorithm is able to handle previous knowledge and therefore either reduce the calculation time or increase the precision of the result. From the presented algorithms only the Structural Equation Method requires previous knowledge to start a calculation. Meanwhile, the Peter-Clark Algorithm can factor in previous knowledge to both reduce the calculation time and increase the precision. The other three algorithms are not able to handle previous knowledge.

4.3.4 Can be applied to BIFROST

This subsection focuses on the application of the presented algorithms on the BIFROST simulation engine as shortly discussed for each algorithm. As the BIFROST simulation engine provides timeseries as input data, the Granger Causality, the Transfer Entropy and the Peter-Clark Algorithm can be applied without further processing the input data.

The Neyman Rubin Causality cannot be applied as the BIFROST simulation engine is not able to collect the pairs of outcomes for each property pair. Finally, the Structural

Equation Method cannot be applied to BIFROST, because BIFROST does not provide previous knowledge which is required for the Structural Equation method.

Summarized we can say the input data of BIFROST enables the application of three algorithms on the BIFROST simulation engine and disqualifies the other two. The three implemented algorithms are the Granger Causality (Section 6.3), the Transfer Entropy (Section 6.4 and 6.5) and the Peter-Clark Algorithm (Section 6.6).

CHAPTER 5

Implementation

In this chapter we investigate RQ4 in terms of the implementation of the causality acquisition algorithms. Therefore, this chapter presents a framework which is able to acquire causality from the BIFROST simulation engine. The framework presents a generic interface which allows the uniform use of different causality acquisition algorithms. In Section 5.1 an introduction to the framework and the functionality is presented. Afterwards, in Section 5.2, the architecture of the framework and the workflow of the framework is described. The chapter concludes with Section 5.3 which describes a set of metrics. These metrics are provided by the framework and can be used to evaluate the acquired causal relations. These metrics are a contribution of the thesis.

5.1 Introduction

This section presents the implementation to acquire and evaluate causality relations from the BIFROST simulation engine. To perform this acquisition and evaluation tasks, a framework is developed. The framework has the ability to acquire pairwise causality relations of properties for each collected timeseries from the BIFROST simulation engine. It further provides different metrics to evaluate the correctness and performance of different algorithms. The framework is further designed to easily extend the currently existing acquisition algorithms with other causality acquisition algorithms. Currently the implementation contains three different causality acquisition algorithms:

- Granger Causality: The framework uses the implementation provided by [28] and uses the implementation to pairwise compare two different timeseries.
- Transfer Entropy: For the Transfer Entropy the JIDT libary [29] is used. The implementation uses the Kernel and the Kraskov implementation to pairwise compare timeseries.

- Peter-Clark Algorithm: The implementation provided by Tetrad [30] is used in the framework to build a directed acyclic graph based on all timeseries provided by BIFROST.

5.2 Architecture

The framework is based on a Java interface, which can be implemented to use and compare different causality acquisition algorithms. Furthermore, the framework uses a REST interface to get commands and provide results. It always takes a list of all timeseries provided by BIFROST, in a specific format as input and provides a list of pairwise causal relations and a list of calculated metrics as output.

The framework is further built to collect data from a graph database connected to an expCPS module of BIFROST, presented in [AES+20]. The expCPS module is able to provide explanations for events happening in BIFROST, while the framework is created to find causal relations used by the expCPS module to build these explanations.

An example for a successful run of the framework is shown in Figure 5.1, where the following steps are presented:

1. The client (user) requests data from a Settlement ($FI, FT, P, PT, TS, FIFT, PPT, FIP$) from the framework through the REST-Interface.
2. The framework forwards the request to the GraphDB.
3. The GraphDB sends the requested data to the framework.
4. The framework sends the data back to the client.
5. The client uses the received data to request the calculation of causality relations and metrics in the framework through the REST-Interface.
6. The framework forwards the received data about the settlement to the requested CAA.
7. The CAA provides the results (a set of causal relations and a set of quality metrics, described in Section 5.3) to the framework.
8. The framework sends the received results back to the client.

5.3 Metrics

In this section the different metrics provided by the framework are presented. Each metric is calculated based on the provided data and the chosen algorithms.

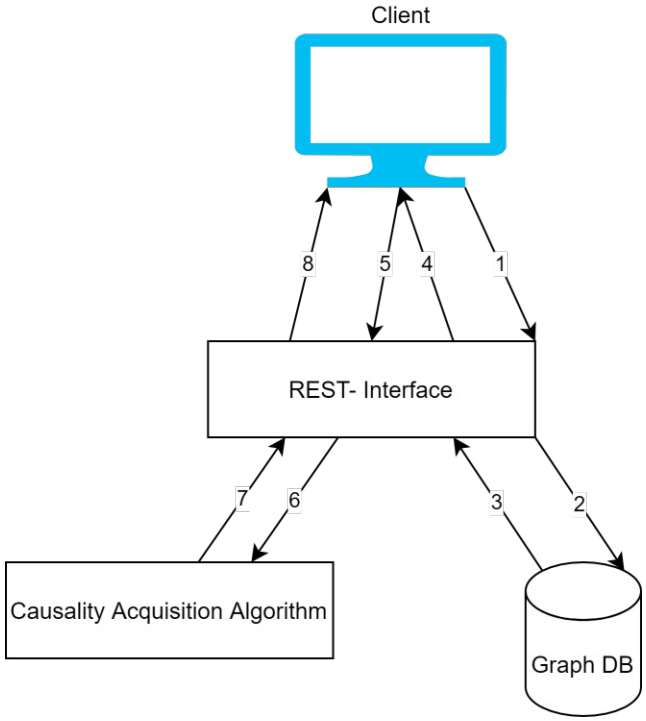


Figure 5.1: Example for a successful run in the framework

5.3.1 Certainty based on object information

The value for this metric is based on a comparison of similar relations between features of interest in a Settlement. Therefore, it takes the result of a Causality Acquisition Algorithm and calculates the percentage of found similar relations compared to possible similar relations, for each found causal relation. Similar relations are defined by the Feature Type and the Property Type. An example for this metric is shown in Figure 5.2 where a transformer (T) is connected to two houses (features of interest, H1, H2) of the same Feature Type. In this example, if a causality relation is found between a Property p_1 of house H1 and a Property p_t of the transformer T and the same causality relation exists between Property p_2 of house H2 and p_t this metric evaluates to 100%. If the relation between p_2 and p_t does not exist, the metric evaluates to 50%.

In the framework, the similarity between relations is based on Feature Types of Features of Interest and Property Types of Properties corresponding to Features of Interest. Therefore there are two cases for the similarity between two causal relations c_1, c_2 of C:

- The cause and the effect of c_1 have the same Property Type and the same Feature Type as c_2. Therefore, if the cause of c_1 has the Property Type *loading_level* and the Feature Type $transformer$, c_2 has to have the Property Type *loading_level*

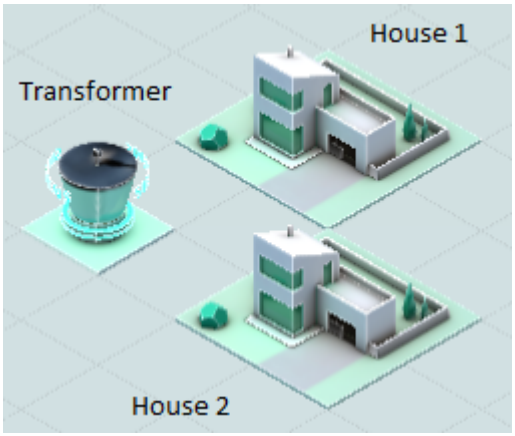


Figure 5.2: Example of BIFROST settlement for illustrating the metrics of "Certainty by object information"

and the Feature Type $transformer$. The same has to be true for the effect of c_1 and c_2 for the causal relations to be counted as similar.

- The cause and the effect of c_1 are the same Feature of Interest fi_1 and the cause and the effect of c_2 are the same Feature of Interest fi_2, where fi_1 and fi_2 have the same Feature Type. Further the Property Type of the cause of c_1 and c_2 are the same and the Property Type of the effect of c_1 and c_2 are the same. For example there are two $residential\_houses$ (Feature of Interest), which have the same Properties with the Property Types $loading\_level$ and $active\_power$. Therefore the causal relations $c_1(loading\_level_1, active\_power_1)$ and $c_2(loading\_level_2, active\_power_2)$ are similar.

Formally the Certainty by object information for a single causal relation ($C_o(c)$) is given in Formula 5.1, where m represents the number of similar causal relations of c. The parameter $\exists c_n$ can either be 1 or 0 if the causal relation was found by the algorithm or not. 1 indicates the success of finding the relation, 0 indicates the missing of the causal relation in the calculation result, as seen in Formula 5.3. Therefore, Formula 5.1 takes the amount of found similar causal relations and divides it through the amount of possible similar causal relations. Applied to the example in Figure 5.2, this would mean, if a relation between p_1 of house 1 and p_t pf the transformer is found, but not between p_2 of house 2 and p_t, m would be two, as there is the possibility of two causal relations and $\sum_{n=1}^{m} \exists c_n$ would be one, as one causal relation is found. Therefore, $C_o(c)$ results to 50% as described before. Furthermore, Formula 5.2 represents the Certainty by object information over a whole system, where j indicates the amount of unique possible, similar causal relations and $C_o(i)$ indicates the Certainty by object information of a single relation, calculated in Formula 5.1.

$$C_o(c) = \frac{\sum_{n=1}^{m} found(c_n)}{m} \tag{5.1}$$

$$C_o = \frac{\sum_{i=1}^{j} C_o(i)}{j} \tag{5.2}$$

$$found(c) = \begin{cases} 1 & \text{if } c \text{ is in } R_s \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

5.3.2 Certainty by ensemble based learning

This metric aims to provide a value of the comparison between the result of different causality acquisition algorithms. To calculate the value, the framework takes the result of each acquisition algorithm and compares the found causal relations. The provided percentage provides the certainty if a causal relation is real, and not a false positive of an algorithm based on the appearance of the relation in each of the provided algorithms. Therefore, if each algorithm has found the causal relation c_n, the certainty is 100%. For each algorithm, which does not find the causal relation c_n the percentage drops.

Formally the Certainty by ensemble based learning C_e for a single causal relation ($C_e(c)$) is given in Formula 5.4, where m is the number of different CAAs used. Therefore Formula 5.4 represents the Certainty by ensemble based learning of a single causality relation, where $\exists c_n$ results to either 0 or 1. It results to 0 if the causal relation is not found by the system and results to 1 if the causal relation is found, as seen in Formula 5.3. To calculate the mean of the Certainty by ensemble based learning of found causal relations, Formula 5.5 is applied, where j is the cardinality of R_s.

$$C_e(c) = \frac{\sum_{n=1}^{m} found(c_n)}{m} \tag{5.4}$$

$$C_e = \frac{\sum_{i=1}^{j} C_e(i)}{j} \tag{5.5}$$

5.3.3 Amount of found causal relations

The last provided metric is the amount of found causal relations, which provides the number of causal relations found by a single algorithm. Formally the amount of causal relations is the cardinality of R_s, that is $|R_s|$.

The implementation and metrics described in this chapter enabled the systematic evaluation of the various CAAs, as detailed in Chapter 6.

CHAPTER 6

Evaluation of Causality Acquisition Algorithms

In this chapter we continue to investigate RQ4. Each implemented algorithm (Granger Causality, Transfer Entropy and Peter Clark Algorithm) is evaluated based on the metrics described in Section 5.3 and following a *controlled experiment methodology*. As there is currently no gold standard of causal relations in the BIFROST simulation engine, it is impossible to compare the result of the framework to a standard. Section 6.1 provides an overview of the experiments used to evaluate the algorithms. In Section 6.2 datasets are presented, which are used to evaluate the different algorithms. The chapter continues with Sections 6.3 – 6.6, where each section describes the evaluation of a single algorithm based on the experiments described in Section 6.1. These evaluations are a contribution of the thesis. After evaluating each algorithm on its own, Section 6.7 briefly compares the algorithms with each other, which is another contribution. Finally, Section 6.8 summarizes the results of the evaluation and provides an overview of each result of an experiment.

6.1 Evaluation Goal and Experiment Overview

In this chapter we conduct two controlled experiments for each implemented Causality Acquisition Algorithm (CAA) with the following goals:

- **Goal 1: Optimal Parameterization.** We aim to identify which parameter settings lead to the optimal performance of a CAA in terms of runtime, C_o and number of found causal relations. As we only aim to investigate the parameters of the algorithm, we use a dataset with average characteristics. Therefore, we use Dataset 4, which has a Measurement frequency of 15 minutes, 3 different Feature Types and 100 datapoints/property (as explained in more detail in 6.2).

Experiment Nr.	Evaluated Algorithm	Aim of the experiment	Used Datasets	Section
Experiment 1	Granger Causality	Optimal Parameterization	4	6.3.2
Experiment 2	Granger Causality	Influence of the Dataset	1 - 10	6.3.3
Experiment 3	Transfer Entropy with a Kernel Estimator	Optimal Parameterization	4	6.4.2
Experiment 4	Transfer Entropy with a Kernel Estimator	Influence of the Dataset	1 - 10	6.4.3
Experiment 5	Transfer Entropy with a Kraskov Estimator	Optimal Parameterization	4	6.5.2
Experiment 6	Transfer Entropy with a Kraskov Estimator	Influence of the Dataset	1 - 10	6.5.3
Experiment 7	PC algorithm	Optimal Parameterization	4	6.6.2

Table 6.1: List of experiments performed in the thesis

- **Goal 2: Dataset Influence.** We aim to study the influence of the dataset characteristics on the results of each implemented CAA. To that end, we identify a number of dataset characteristics in Section 6.2 and define a number of individual datasets based on variations of these characteristics (see Table 6.2). Then we test the algorithm with the optimal parameter values determined in the *Optimal Parameterization* experiment on these diverse datasets to capture the influence of individual characteristics on algorithm performance in terms of the runtime of the algorithm, Certainty by object information (C_o) and the number of results in terms of the derived causality relations.

Each experiment is a *controlled experiment* following the methodology introduced by Wohlin [WRH+12].

Manual Inspection During each experiment, the experimenter also manually inspects the result set R_s. As the known causal relations of the used expCPS are unknown, there are only two checks which can be manually performed. These checks are:

- Each causality relation, where the effect is the weather and the cause is not the weather, has to be false positive, as we assume, no building can influence the weather.

Dataset	FTc	Structure	Mf	no. of records / property
1	2 Feature Types	Figure 6.1	15 min	100
2	3 Feature Types	Figure 6.2	15 min	10
3	3 Feature Types	Figure 6.2	1 min	100
4	3 Feature Types	Figure 6.2	15 min	100
5	3 Feature Types	Figure 6.2	60 min	100
6	3 Feature Types	Figure 6.2	120 min	100
7	3 Feature Types	Figure 6.2	15 min	200
8	8 Feature Types	Figure 6.4	15 min	100
9	3 Feature Types	Figure 6.2	15 min	500
10	5 Feature Types	Figure 6.3	15 min	100

Table 6.2: List of parameters of used datasets

- The experimenter can check for logical implications. For example, if the *temperature_base* increases the *current_temperature* increases as well. Therefore the CAA should find this causal relation.

6.2 Evaluation Datasets

The evaluation is based on 10 datasets, which have different combinations of characteristics. These datasets are described in this section. The datasets and their characteristics are listed in Table 2.5 and the BIFROST settlements which generate these datasets are illustrated in Figures 6.1-6.4. Each dataset is extracted based on the predefined parameters, but as the simulation differs based on random aspects, two datasets taken directly after each other with the same parameterization, could yield different results.

6.2.1 Feature Type complexity

The first metric used to differentiate datasets is the **Feature Type complexity** FTc, where the amount of used Feature Types changes. Therefore, a more complex settlement contains more different Feature Types. A Settlement with a low diversity across the placed Feature Types has an overall lower FTc than Settlements with high diversity across placed Feature Types. This is based on the difference in causal relations, which can be found. In an environment with a low FTc the amount of different relations is lower, because between the same type of Feature Types the same relations exist. In comparison, a highly complex environment has a variety of different causal relations. The number of Features of Interest of the same type does not matter, due to the construction of C_o, which is used as a quality metric. Therefore, this number is chosen randomly. For the evaluation four different settlements are used.

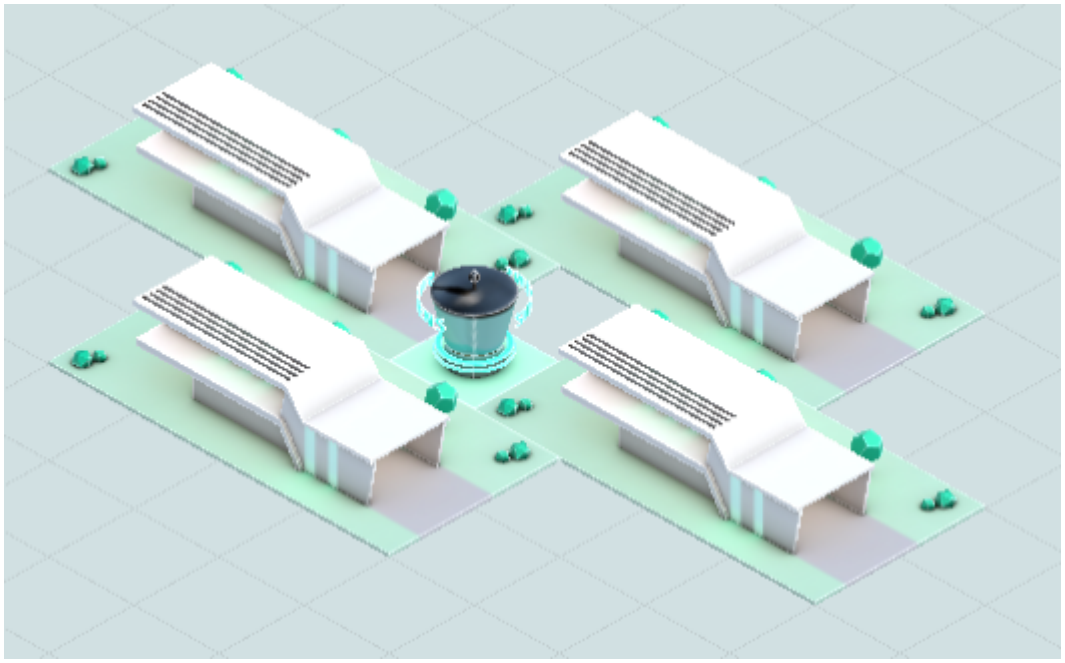


Figure 6.1: Settlement with 2 Feature Types (4 residential buildings, 1 transformer)

- Settlement 1: A Simple Settlement shown in Figure 6.1 shows a setup with five Features of Interest (two Feature Types), which are a Transformer and four residential buildings.
- Settlement 2: The Settlement shown in Figure 6.2 shows a setup with 15 Features of Interest (three Feature Types), which are nine residential buildings, two apartment buildings and a transformer.
- Settlement 3: The Settlement shown in Figure 6.3 shows a setup with 12 Features of Interest (five Feature Types), which are three apartment buildings, two single family houses with a pool, three multi floor buildings, two single-family houses and a transformer)
- Settlement 4: A complex Settlement shown in Figure 6.4 shows a setup with 21 Features of Interest (nine Feature Types), which are four residential buildings, five apartment buildings, a single family house with pool, three multi floor buildings, two single family houses, two single family houses with a garage, two single family houses with a parking space and two transformers)

6.2.2 Measurement frequency

Another important metric is the frequency of individual measurements in a timeseries, called **Measurement frequency** Mf, where the time between the recordings of datapoints is set. This is an important metric as some algorithms can heavily depend on Mf

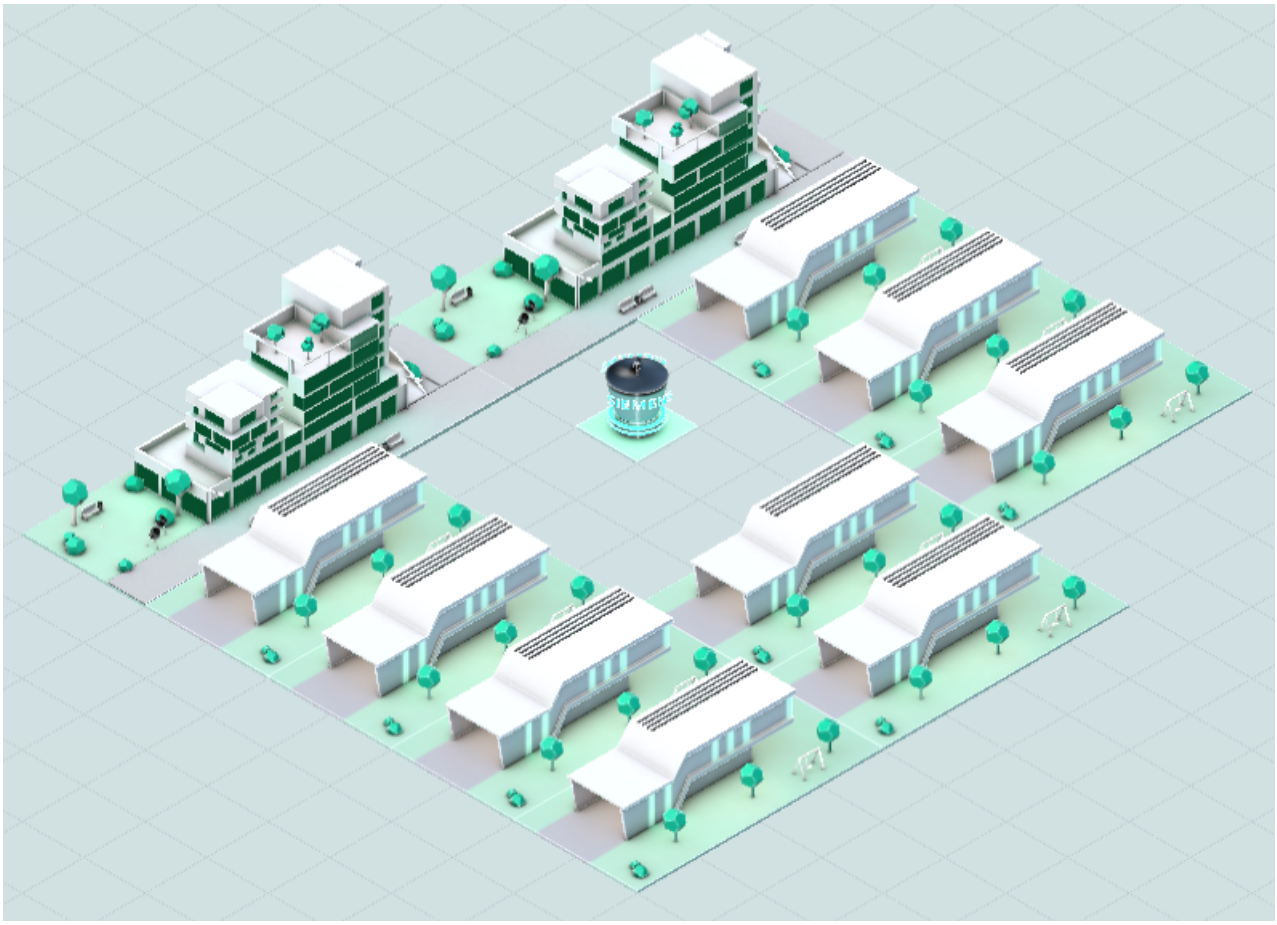


Figure 6.2: Settlement with 3 Feature Types (9 residential buildings, 2 apartment buildings, 1 transformer)

and sometimes even have an own parameter to change the metric (skipping entries in a timeseries). Formally Mf is given in Formula 6.1, where the difference in time between two Datapoints is calculated. For the evaluation we use Measurement frequencies between one minute and two hours. The used frequencies are predefined by BIFROST, as the simulation engine only provides timesteps of 1 minute, 15 minutes, an hour and two hours.

$$Mf = ts(t)[t] - ts(t-1)[t] \tag{6.1}$$

6.2.3 Number of Datapoints

The last metric considered is the **number of datapoints** which are recorded for each timeseries of a dataset. The goal is to find a fitting size for the algorithm to reduce calculation time and increase the quality and accuracy of the found causal realtions.

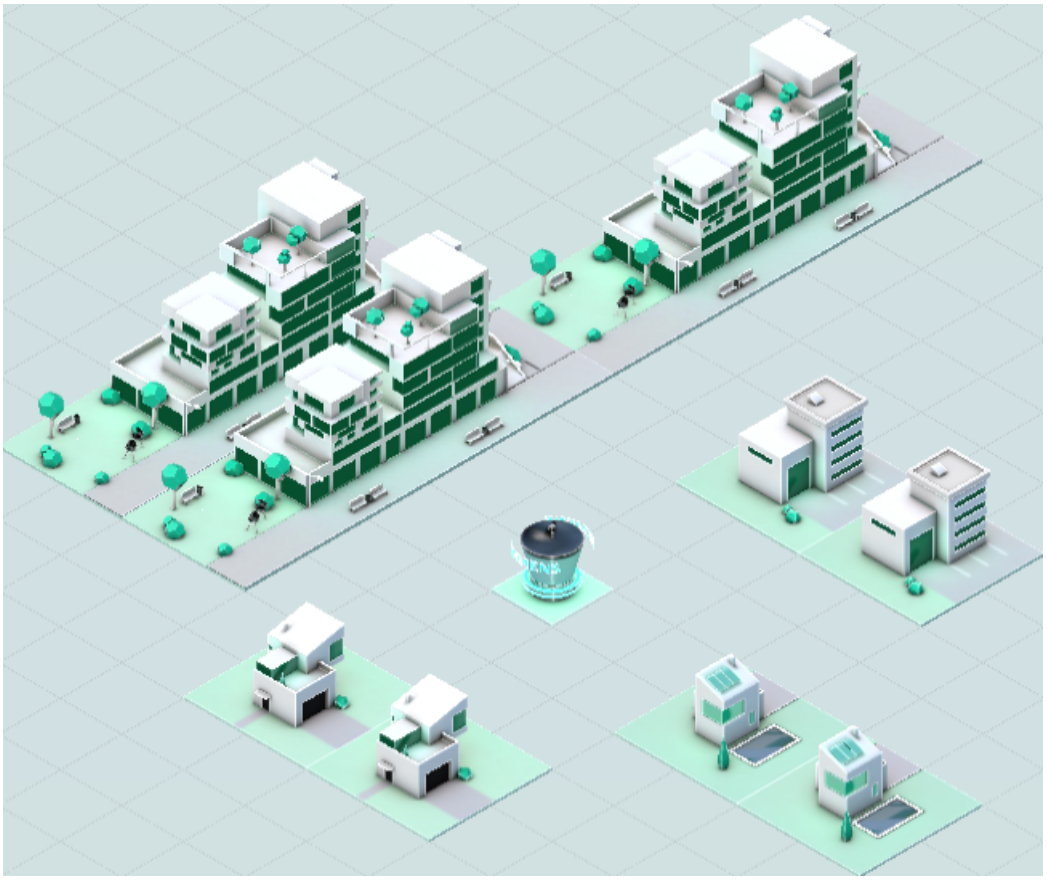


Figure 6.3: Settlement with 5 Feature Types (3 apartment buildings, 2 single family houses with pool, 3 multi floor buildings, 2 single-family houses, 1 transformer)

Formally the number of datapoints is the cardinality of the used timeseries. For the evaluation we use datasets with 10 to 500 datapoints.

6.3 Granger Causality

The Granger Causality as described in Section 4.2.1 compares two different timeseries and tries to find a causal relation between the Properties measured by the timeseries. For the implementation we use the library provided at [28].

6.3.1 Parameters

The Granger causality can be parameterized by two different parameters: the **lag size** ls and the **critical value** cv. ls is similar to the Measurement frequency Mf as it describes the time between the current value and the historic one, which is compared to the current

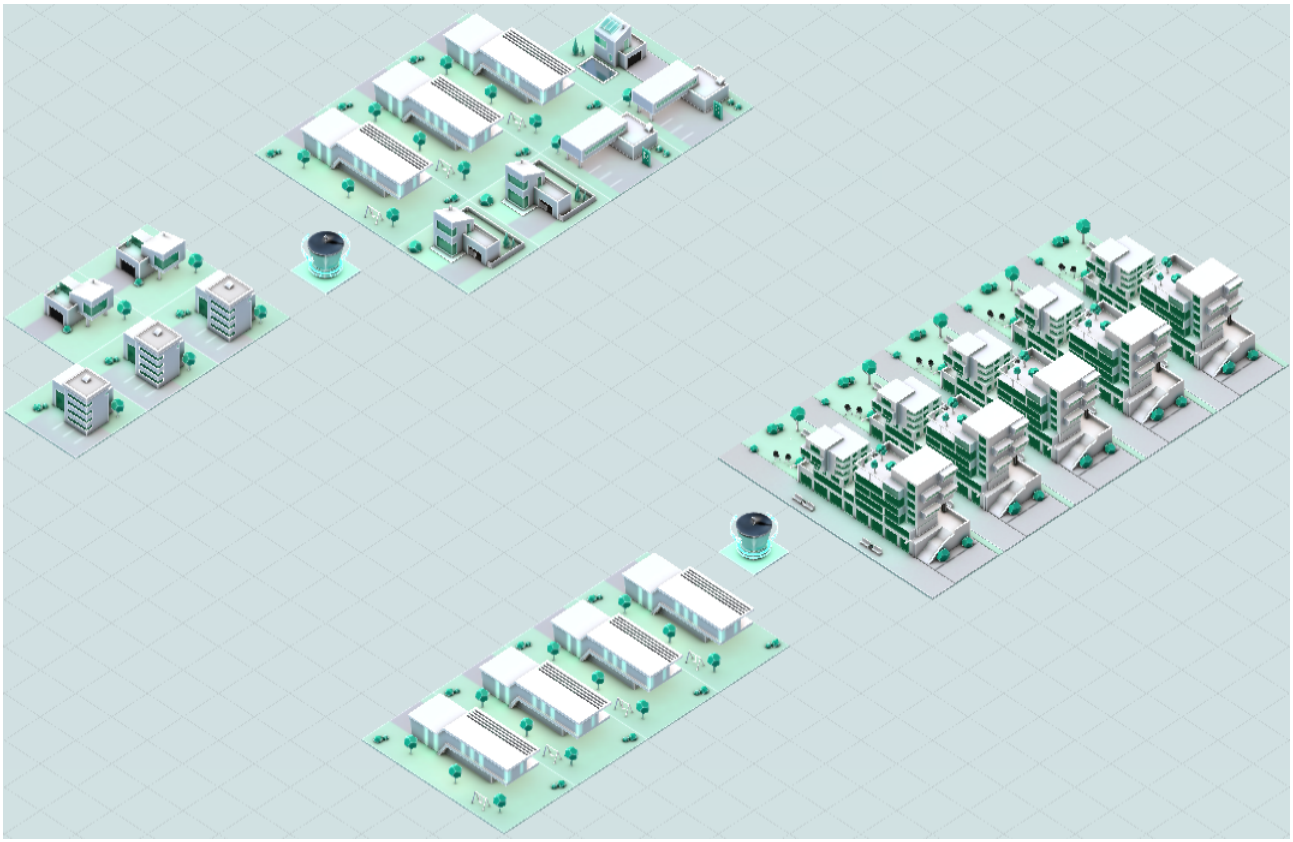


Figure 6.4: Settlement with 8 Feature Types (4 residential buildings, 5 apartment buildings, 1 single family house with pool, 3 multi floor buildings, 2 single family houses, 2 single family houses with a garage, 2 single family houses with a parking space, 2 transformers)

one. ls is given as the number of values which are skipped. Therefore, the time lag between the current value and the historic value is given by $Mf * ls$.

The other value which can be parameterized in the algorithm for the Granger Causality is the critical value cv, which is the confidence interval. If the calculated p-value is bigger than the defined cv, the comparison of the two timeseries will be defined as causal related.

For the evaluation we use different confidence intervals between 60 and 90 percent. The used lag size is between one and four values, where the evaluation has to take the relation towards the Measurement frequency into account.

6.3.2 Experiment 1: Optimal Parameterization

In this section we evaluate the parameterization of the Granger Causality using a controlled experiment.

Aim

This experiment aims to evaluate the influence of the different parameters on the result set. Therefore, we aim to investigate the difference in quality metrics (runtime, C_o, no. of results) while changing parameters of the algorithm. Furthermore, we aim to find an optimal setting of parameters in terms of ls and cv.

Hypothesis

As we follow the experimentation approach presented in [CI11], we start by defining the *null hypothesis* h_0 and the *alternative hypothesis* h_a for the experiment. If the null hypothesis is rejected, the alternative hypothesis will be accepted. Therefore, we define the alternative hypothesis as follows:

h_a: The parameterization of the Granger Causality influences the result of the algorithm and therefore an optimal set of parameters can be defined.

Context

The experiment is conducted in the BIFROST simulation engine and for each set of parameters a new test run is started. Therefore, each test run is a *subject* of the experiment. For each test run the algorithm has new parameters assigned, while the dataset characteristics stay the same. Further the aim is to evaluate the algorithm, which is the *object* of the experiment. As we only aim to investigate the parameters of the algorithm in this experiment, we use a dataset with average characteristics. Therefore, we use dataset 4, which has $Mf = 15$ minutes, three different Feature Types and 100 datapoints/property.

Procedure

For the execution of the experiment, the evaluator conducts successive test runs, while always changing the combination of parameters. Therefore, the evaluator starts the first test run with the lowest possible setting for each parameter and afterwards runs the algorithm with each possible combination of parameters. After performing the testrun, the evaluator manually checks the results for abnormalities and documents the setting of the test run and the results.

Variables

According to the experimentation standard, different independent and dependent variables are defined for each experiment. Each independent variable is a variable, which can be influenced by the evaluator, while dependent variables result from the execution of an experiment. In our case we can define the different parameters of the algorithm as independent variables. Therefore, the only *independent variable* in this experiment is cv, as the second parameter, ls, is directly related to the Measuring frequency Mf of the properties in the timeseries of the dataset and therefore cannot be changed to keep Mf stable. We therefore set $ls = 1$ for this experiment. Finally, the *dependent variable* in the experiment is the *performance*, which is measured by a combination of the *runtime* of the algorithm, the *Certainty by object information* (C_o) and the *no. of results*.

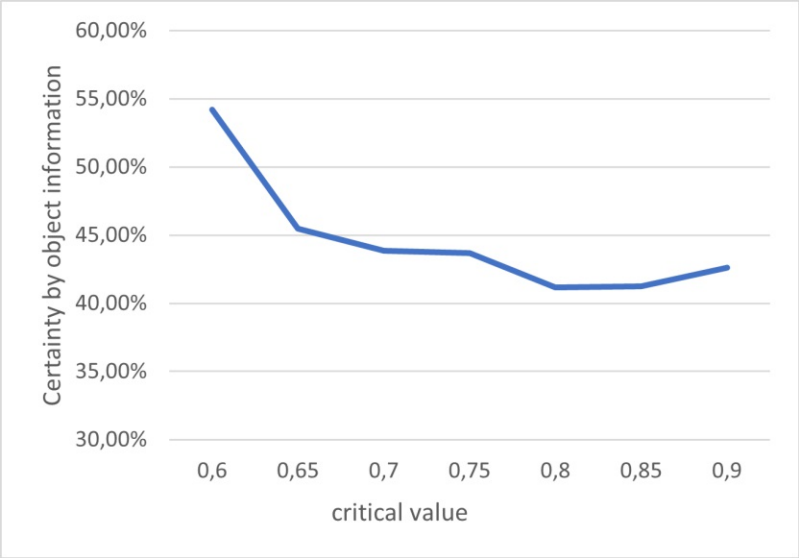


Figure 6.5: Granger Causality: Evaluation of Certainty by object information

Evaluation

To reject the null hypothesis, we have to show the difference of the results depending on the *independent variables* and how the performance of the algorithm changes with the change of parameters.

As mentioned earlier, the performance has three different values which should be optimized. These values are the *runtime*, C_o and the *no. of results*. For the Granger Causality we can neglect the runtime as every test run has a runtime below three seconds. Therefore, the two main quality metrics are the no. of results and C_o.

In Figure 6.5 and 6.6 C_o and the *no. of results* over an increasing cv is shown. As expected, the *number of results* is indirectly proportional to cv, as an increasing cv reduces the number of allowed results. Further, we can see C_o keeps stable after an initial drop.

According to these results, the optimal setting for the **critical value** would be 0,6 because it has the highest C_o and the most results. After a manual inspection of the results we found a high number of false positive results in the results set with $cv = 0,6$. Therefore, the optimal setting is $cv = 0,7$, as it has the highest number of results with a similar C_o compared to other measurements. The existence of this setting also rejects the null hypothesis and therefore accepts the alternative hypothesis.

Examples of Identified Causality Relations (R_s)

To provide a qualitative insight into the obtained causality relations we present portion of R_s in:

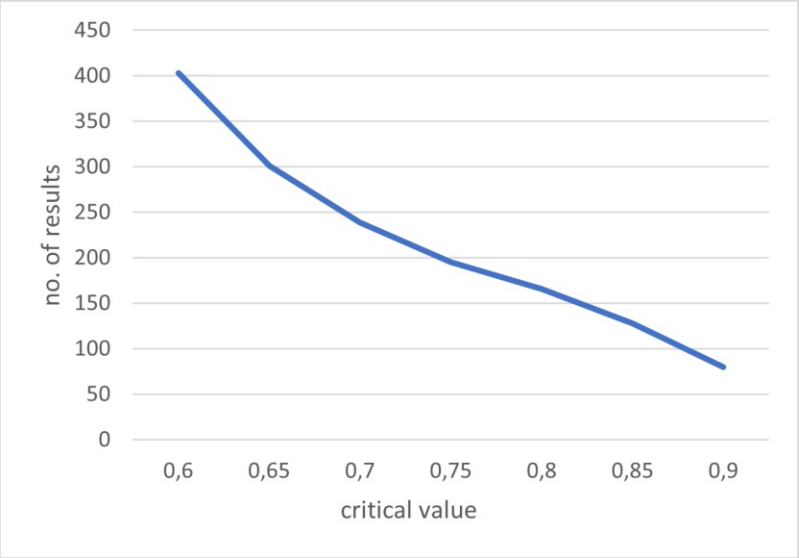


Figure 6.6: Granger Causality: Evaluation of number of results

- Table 6.3 showing 7 of the 239 relations obtained with the setting $ls = 1$ and $cv = 0,7$. The table shows the fi/p pairs for the cause and the effect of causal relations from R_s as well as the value of each relation in terms of the C_o metric. For example, the first row shows the causal relation between the Property $TRAFO - SECONDARY - VOLTAGE$ of the Feature of Interest $TRAFO - BUILDING$ (cause) and the Property $VOLTAGE - 3P$ of the Feature of Interest $RESIDENTIAL - MULTI - LARGE$ (effect). This causal relation was verified for all pairs of this fi/p combination ($C_o = 100\%$).

- Figure 6.7 depicting C_o for each causal relation of R_s with the optimal parameterization. Each axis in the figure shows all Properties available in Settlement 2. Each square provides the color coded C_o for the causal relation between the cause (y-axis) and the effect (x-axis), where violet shows $C_o = 0\%$ and green shows $C_o = 100\%$ For example the causal relation between the Property $CLOUDCOVER - BASE$ of the fi $AIRSHIP$ (cause) and the Property $VOLTAGE - ANGLE - 3P$ of the fi $RESIDENTIAL - MULTI - MEDIUM$ (effect) in the top right corner, has $C_o = 100\%$, as the square is green. Intuitively, we found that $CLOUDCOVER - BASE$ has an effect on $VOLTAGE - ANGLE - 3P$ of $RESIDENTIAL - MULTI - MEDIUM$, and this causal relation was verified for all such houses in the settlement. If we swap the cause and the effect, we can see $C_o = 0\%$ in the square in the bottom left corner, as this square is purple. A $C_o = 50\%$ can be seen for each white square in the figure.

  In Figure 6.7 we can see the Granger Causality only found a small amount of causal relation with a high C_o. The found causal relations are related to the Property $CLOUDCOVER - BASE$ of the fi $AIRSHIP$, where this Property is found

Cause		Effect		Certainty by object information
Feature of Interest	Property	Feature of Interest	Property	
TRAFO-BUILDING	TRAFO-SECONDARY-VOLTAGE	RESIDENTIAL-MULTI-LARGE	VOLTAGE-3P	100%
AIRSHIP	SUN-ALTITUDE	AIRSHIP	DIRECT IRRADIANCE	100%
RESIDENTIAL-MULTI-LARGE	ACTIVE-POWER	TRAFO-BUILDING	REACTIVE-POWER	50%
RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	RESIDENTIAL-MULTI-LARGE	ACTIVE-POWER-3P	15%
AIRSHIP	TEMPERATURE-BASE	AIRSHIP	CURRENT-TEMPERATURE	100%
AIRSHIP	CLOUDCOVER-BASE	RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	78%
RESIDENTIAL-MULTI-LARGE	VOLTAGE-3P	TRAFO-BUILDING	REACTIVE-POWER-3P	100%

Table 6.3: Sample of R_s of the Granger Causality

as cause for multiple effects. The causality acquisition algorithm only found six causal relations between buildings of the settlement with $C_o = 100\%$. Further, the algorithm found a lot of causal relations with a low C_o (violet to white squares), which indicates the possibility of a false positive result.

6.3.3 Experiment 2: Influence of Dataset Characteristics

This Section describes an experiment to find the behavior of the algorithm for changing dataset characteristics.

Aim

In this experiment we aim to investigate the behavior of the algorithm for changing dataset characteristics. Therefore we evaluate how, the result set changes for datasets with different dataset characteristics.

Hypothesis

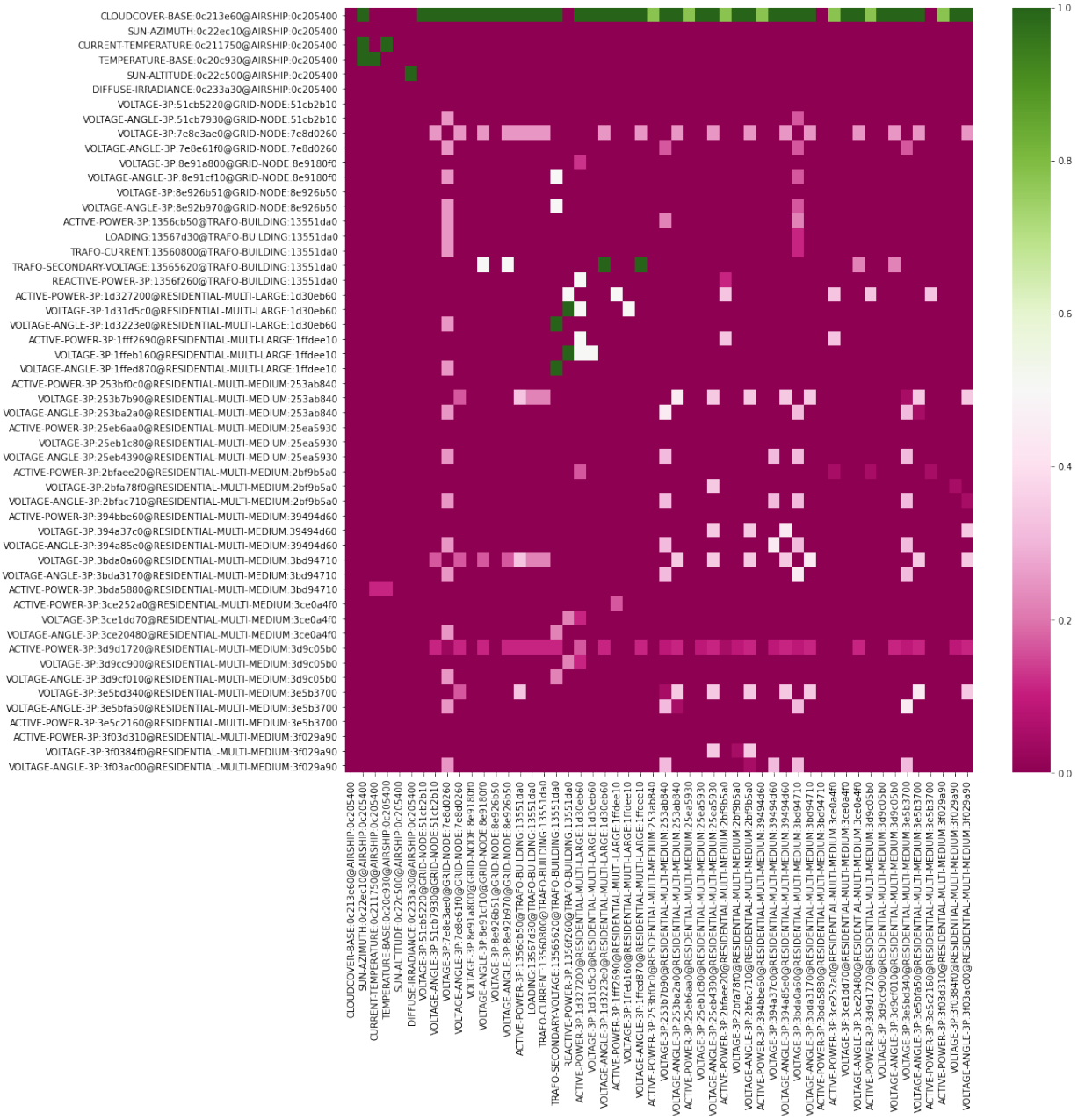


Figure 6.7: Granger Causality: Matrix representation of R_s of the Certainty by object information. Legend: The color scale indicates the C_o from violet (0%) to green (100%)

To conduct this experiment, we create three hypotheses, which are evaluated in the following:

h_{an}: The number of records/property in a dataset influences the result of the Granger Causality and an increase or decrease in the number of records/property changes the result.

h_{ac}: FTc of a dataset influences the result of the Granger Causality and therefore a changing FTc changes the result.

h_{af}: Mf of a dataset influences the result of the Granger Causality and therefore a changing Mf changes the result.

Context

The experiment is conducted in the BIFROST simulation engine and for each used dataset a new test run is started. Therefore, each test run is a *subject* of the experiment. For each test run the dataset changes one characteristics, while the parameterization of the algorithm stays the same. Further the aim is to evaluate the result of the algorithm, therefore the algorithm is the *object* of the experiment. In this experiment we aim to evaluate, how the different characteristics of a dataset influence the result of the Granger Causality algorithm. Therefore, we use the optimal parameterization ($cv = 0,7$, $ls = 1$) found in Section 6.3.2 as a setting for the algorithm.

Procedure

For the execution of the experiment, the evaluator conducts successive test runs, while always changing the dataset characteristics. Similar to the previous experiment we start with a dataset with average characteristics. The starting dataset is Dataset 4. For each hypothesis, the evaluator uses different datasets. For h_{af} datasets 3-6 are used (Mf varies from 1 to 120 min), for h_{an} datasets 2, 4, 7 and 9 (number of datapoints/property varies from 10 to 500) and for h_{ac} datasets 1, 4, 8 and 10 (FTc varies from 2 to 9 Feature Types) are used. These datasets always have one characteristic changed from Dataset 4. After each test run the evaluator manually checks R_s for abnormalities and documents the result and the used dataset.

Variables

The experiment has three *independent variables*, which are changed during the evaluation. These *independent variables* are:

- the Measurement Frequency Mf
- the Feature Type Complexity FTc
- the number of records/property

An overview of all *independent variables* is shown in Table 6.4. The *dependent variable* is the performance, which, for the Granger Causality, is defined through the *Certainty by object information* (C_o) and the *number of results*.

Hypothesis	Independent variable	Scale
h_{an}	number of records/property	10 - 200 records
h_{ac}	FTc	2 - 8 Feature Types
h_{af}	Mf	1 - 120 min

Table 6.4: Independent Variables in the experiment for the influence of the dataset characteristics on the Granger Causality

Evaluation

The first hypothesis we evaluate is the **number of records/property** (h_{0n}). In Figure 6.8 we can see a constant increase of C_o, with the increase of the number of records/property. The number of found results is between 230 and 300, with an outlier at 10 collected datapoints. The distribution can be seen in Figure 6.9. This outlier can be explained with a high number of false positive results, which further causes the low C_o. We can finally conclude: The higher the number of datapoints/properties, the more accurate the calculation of the Granger Causality, while the number of results also increases. Therefore we can reject h_{0n} and accept h_{an}.

Following the finding of maximizing the datapoints/property we can also take a step back and look at the **lag size** ls parameter of the Granger Causality. This parameter is not only directly related to Mf, but also to the number of datapoints/property, as an increase of ls decreases the number of datapoints/property. This behavior results from the algorithm, where an increase of ls skips values and therefore reduces the number of datapoints/property. Therefore, we can conclude $ls = 1$ provides the most optimal results for the Granger Causality algorithm.

In the next step we evaluate the influence of the **Feature Type complexity** FTc of the dataset characteristics on the result of the algorithm (h_{ac}). In Figure 6.10 we can see the decreasing C_o during the increase of FTc. The number of results increases constantly, but this has to be expected as the number of possible causal relations will increase, if more different Feature Types are placed in a settlement. Therefore, we can follow: The higher FTc is, the worse the Granger Causality performs. In the end we can reject h_{0c} and accept h_{ac}.

The last remaining hypothesis to be discussed is h_{af}, which aims to show the influence of the **Measurement frequency** Mf on the result of the Granger Causality. In Table 6.5 the results of the evaluation are shown. Even though $Mf = 1$ minute and $Mf = 120$ minutes seem to be optimal, they cannot be used, as $Mf = 1$ delivers over 2000 results, which is about half of all possible results. Therefore, this result set has to contain a high number of false positive entries. Furthermore, $Mf = 120$ minutes misses whole spikes in the timeseries, as the time between the recordings of the dataset is too long. From the remaining two evaluations, $Mf = 60$ minutes performs better, but even at this time distance there is a possibility of missing a whole spike in a timeseries. Therefore, based on the environment, we recommend using either a $Mf = 15$ or $Mf = 60$ minutes. For

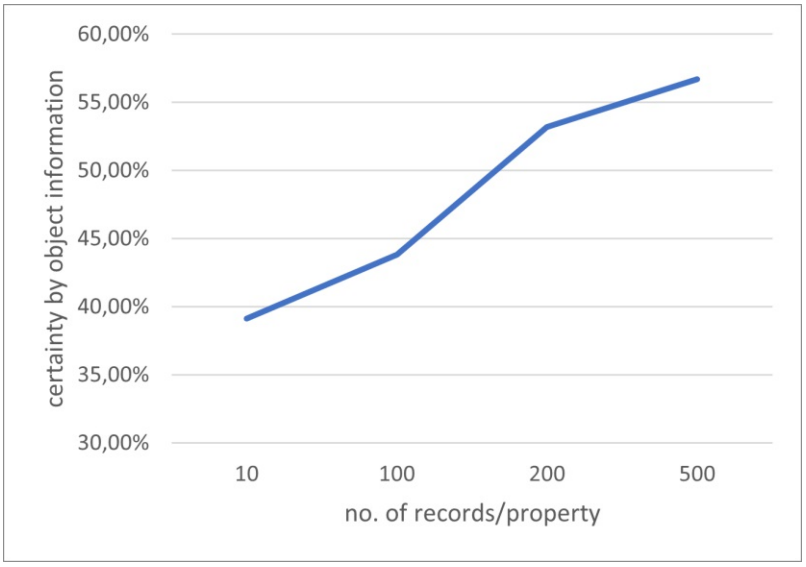


Figure 6.8: Granger Causality: Evaluation of number of records/property (Certainty by object information)

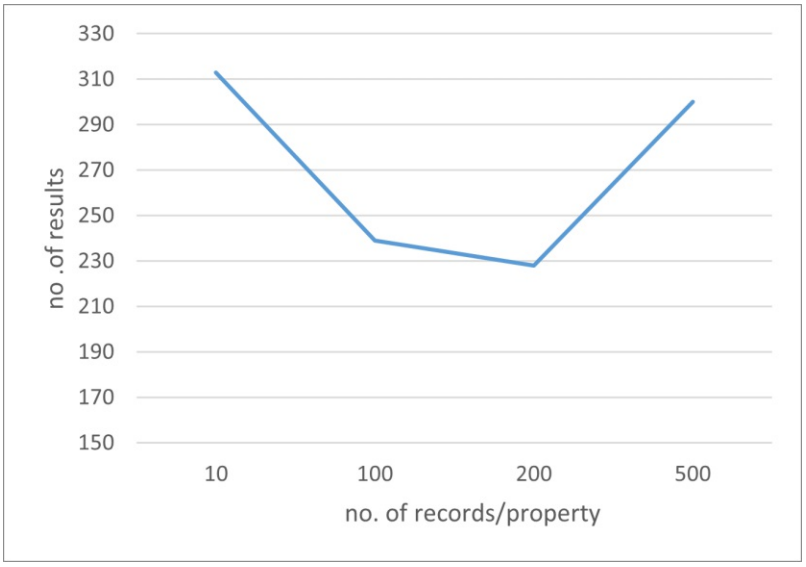


Figure 6.9: Granger Causality: Evaluation of number of records/property (number of results)

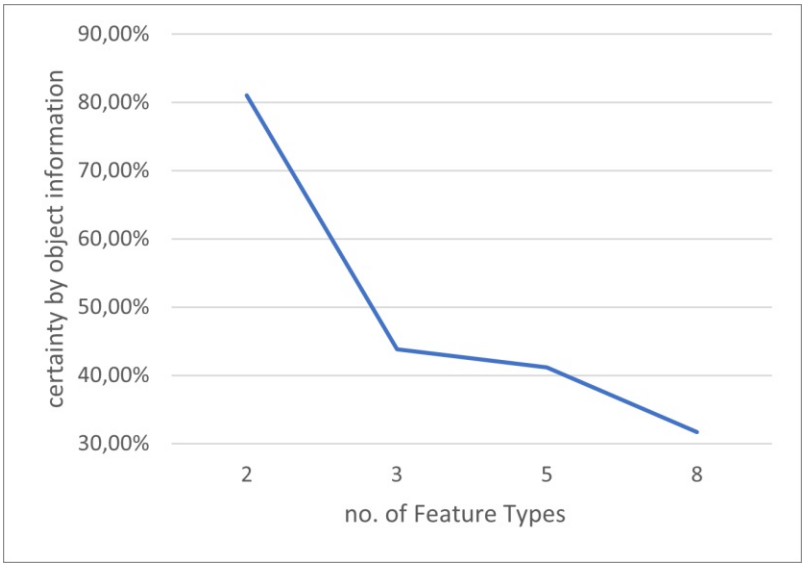


Figure 6.10: Granger Causality: Evaluation of FTc (Certainty by object information)

Dataset	Measurement frequency	Certainty by object information	no. of results
17	1 min	98,62%	2201
18	15 min	43,83%	239
19	60 min	52,59%	164
20	120 min	68,86%	158

Table 6.5: Granger Causality: Evaluation of Mf

the BIFROST simulation engine both frequencies seem to be fine. Finally, we can reject h_{0f} and accept h_{af}.

Conclusions. At the end of this experiment, we can summarize the following results:

- The higher the number of datapoints/property is, the more accurate is the result.
- The more complex a settlement gets, the worse the Granger Causality performs.
- A Measurement frequency between 15 and 60 minutes is optimal for the Granger Causality in the BIFROST simulation engine.

6.4 Transfer Entropy with a Kernel Estimator

The Transfer Entropy aims to find the information flow between two Timeseries, as described in Section 4.2.2. The implementation uses the library provided by [29]

6.4.1 Parameters

The Transfer Entropy with a Kernel Estimator has three different parameters to optimize the result. These parameters are:

- **History length**: The history length *hl* contains the number of values considered from the past. Therefore, a history length of two considers the current value and the previous value to find a possible causal relation.
- **Bandwidth**: This parameter gives the bandwidth *b* of the Kernel Estimator. The influence of this parameter on the result of the calculation is described in Section 4.2.2.
- **Critical value**: The critical value *cv* gives the threshold, in which we accept a causal relation as existing. The result of the Transfer Entropy with a Kernel Estimator is given in nats, which is the unit for the information transfer. Therefore, if the found causal relation has more nats than the critical value, we will accept the relation as causal related.

6.4.2 Experiment 3: Optimal Parameterization

This section describes an experiment to evaluate different combinations of parameters of the Transfer Entropy with a Kernel Density Estimator.

Hypothesis

Similar to previous experiments we follow the same approach of controlled experiments of Wohlin, presented in [WRH+12]. Therefore, our *alternative hypothesis* is:

h_a: The parameterization of the Transfer Entropy with a Kernel Density Estimator influences the result of the algorithm and therefore an optimal set of parameters can be defined.

Context

The context of the experiment is the BIFROST simulation engine, where the evaluator carries out a test run for each possible combination of parameters. Each test run is a *subject* of the experiment. After each test run a single parameter is changed, while the dataset stays the same, namely the average dataset Dataset 4.

Procedure

During the execution of the experiment, the evaluator conducts a test run for each unique combination of parameters. In the first step, the evaluator changes the setting of the algorithm to the new paramters. Afterwards he executes the calculation and documents the results and the used setting of parameters. Further, the evaluator manually checks the result set for abnormalities.

Variables

Hypothesis	Independent variable	Scale
h_a	bandwidth	0,2 - 0,7
h_a	critical Value	0,05 - 0,15
h_a	historic length	1 - 5

Table 6.6: Independent Variables in the experiment for the parameterization of the Transfer Entropy with a Kernel Estimator

As stated in the experiment guidelines, the experiment has different *dependent* and *independent variables*. For this experiment, the parameters, which we aim to evaluate, are the *independent variables*. Therefore, the experiment contains three independent variables:

- the critical value cv
- the bandwidth b
- the historic length hl

An overview is shown in Table 6.6. Further, the *dependent variable* is the performance, which is measured by the *runtime* of the algorithm, the *Certainty by object information* (C_o) and the *no. of results*.

Evaluation

To finish the evaluation we have to accept either the null hypothesis h_0 or the alternative hypothesis h_a. As we further want to find an optimal set of variables, we will take a look at each parameter on his own, while we keep the other parameters at an average or an, already found, optimal value.

We start by showing the influence of the **bandwidth** b on the calculation result of the algorithm. Therefore, we set $cv = 0,1$ and $hl = 1$, as 1 is recommended by Schreiber [Sch00]. In Table 6.7 the results of this setting are shown. We can see the decrease of C_o and the decrease of the no. of results while increasing b.

The high bias of b can be seen for values greater than 0,5 as the no. of results drastically decreases. For these results, C_o loses significance as the amount of found causal relations is low.

The optimal setting for b seems to be 0,2, as it yields the best C_o and the highest no. of results, but after the manual inspection, the high number of results for $b = 0,2$ and $b = 0,3$ is caused by a high number of false positive results. Therefore, $b = 0,4$ is optimal for the BIFROST simulation engine, as it yields the best C_o and the highest number of results without a high number of false positive results.

After showing the optimal setting for b, we continue by exploring the **historic length** hl. As we already found the optimal value for b, we use it for this evaluation and continue using $cv = 0,1$.

Dataset	Bandwidth	Certainty by object information	no. of results
4	0,2	90,72%	1126
4	0,3	80,89%	567
4	0,4	73,65%	335
4	0,5	69,38%	154
4	0,6	22,20%	2
4	0,7	93,59%	42

Table 6.7: Transfer Entropy with a Kernel Estimator: Evaluation of the bandwidth

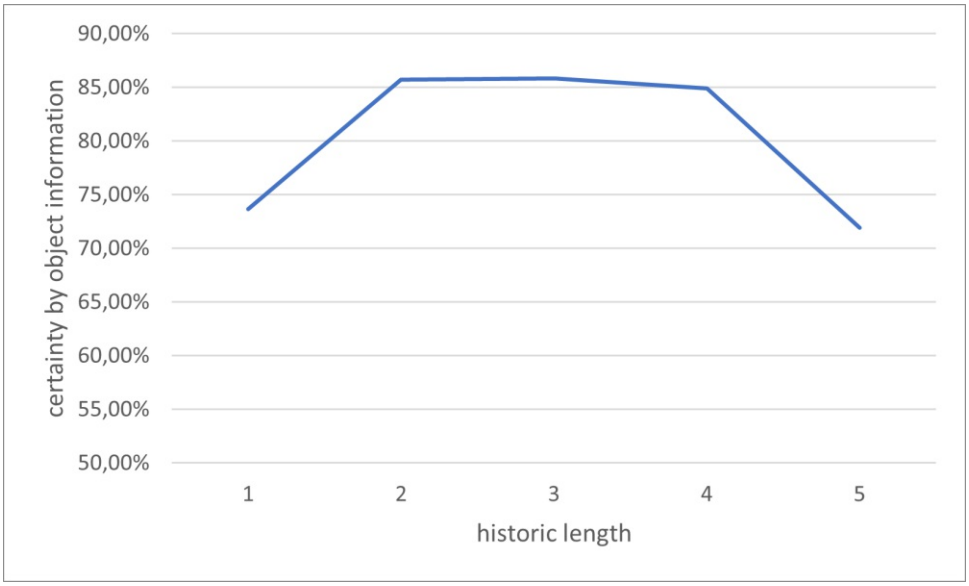


Figure 6.11: Transfer Entropy with a Kernel Estimator: Evaluation of hl (C_o)

During this part of the evaluation it is important to consider the runtime, as each increase of hl increases the runtime exponentially. Therefore, $hl = 1$ takes under a second to calculate, while $hl = 5$ takes over 5 minutes.

In Figure 6.11 C_o over an increasing hl is shown. We can see the peek of C_o is between 2 and 4. The same behavior can be observed in Figure 6.12, where the no. of results over an increasing hl is shown. In hindsight to the runtime, we can therefore follow $hl = 2$ is optimal.

In the last step of the evaluation of this experiment, we evaluate the **critical value** cv. As we already found the optimal parameter for the other two parameters, we use these two for this evaluation.

In Figure 6.13 we can see C_o over an increasing cv. This figure shows a decrease of C_o, while cv increases. The number of results can not be used for this evaluation as this

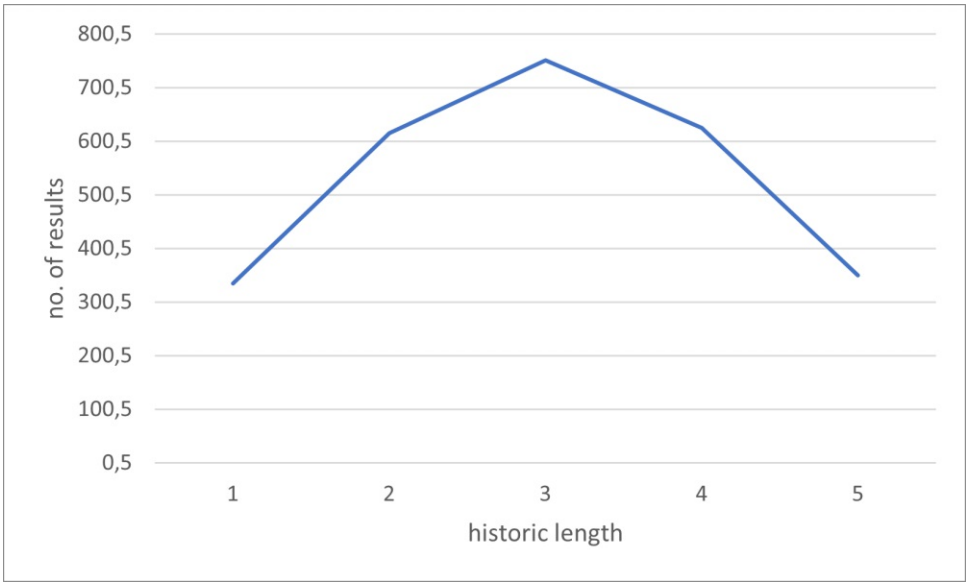


Figure 6.12: Transfer Entropy with a Kernel Estimator: Evaluation of hl (number of results)

metric has to decrease strictly, as an increase excludes more values.

In this evaluation it is important to consider false positive results, as cv is a parameter which aims to exclude these. For a $cv > 0,1$ the number of false positive results is high, and therefore these values cannot be optimal. Therefore $0,1 < cv < 0,15$ is optimal, where both the number of false positive results and C_o constantly decrease.

Conclusions. Summarized, the optimal parameterization for the Transfer Entropy with a Kernel Estimator is $b = 0,4$, $hl = 2$ and $0,1 < cv < 0,15$.

Examples of Identified Causality Relations (R_s)

In this part of the thesis we show an overview of the causal relations, found by the Transfer Entropy with a Kernel Estimator:

- Table 6.8 shows 5 of the 616 causal relations found by the Causality Acquisition Algorithm, with the setting $b = 0,4$, $hl = 2$ and $cv = 0,13$. For example, the Property $ACTIVE-POWER-3P$ of the Feature of Interest $RESIDENTIAL-MULTI-MEDIUM$ (cause) is causally related to the Property $ACTIVE-POWER-3P$ of the Feature of Interest $TRAFO-BUILDING$ (effect), but as $C_o = 34\%$ this causal relations was only identified for 1/3 of all $RESIDENTIAL-MULTI-MEDIUM$.
- Similar to the results of the Granger Causality in Section 6.3.2, the results of the Transfer Entropy with a Kernel Estimator is shown in Figure 6.14. In this result set, we can see a lot of causal relations related to the weather (FI:

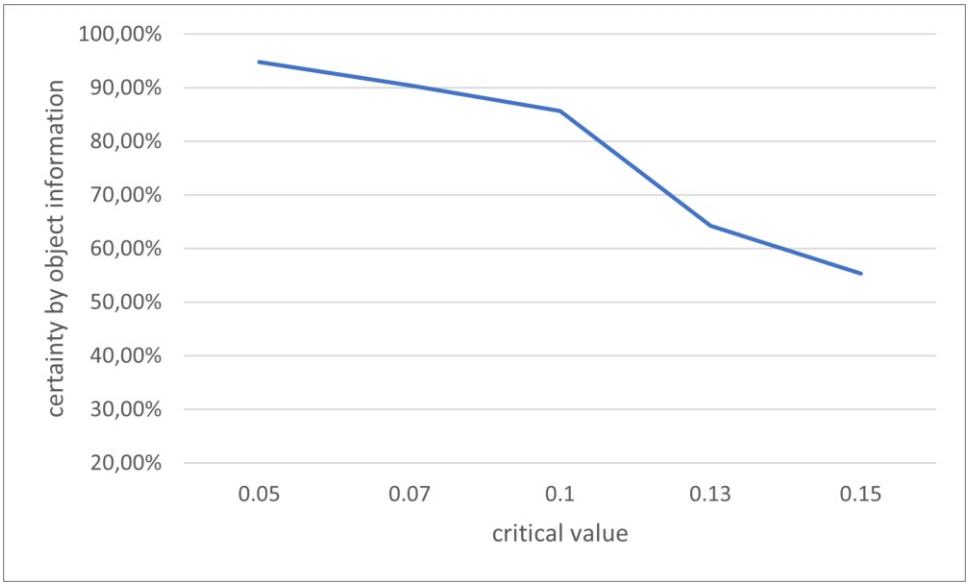


Figure 6.13: Transfer Entropy with a Kernel Estimator: Evaluation of cv (C_o)

AIRHSIP). Further, we can see a high amount of causal relations between the Properties $CLOUDCOVER-BASE$ and both temperature properties($CURRENT-TEMPERATURE$, $TEMPERATURE-BASE$) and every other Property in the Settlement. From a logical point of view, these causal relations seem to be false positive results. Compared to the results of the Granger Causality in Secion 6.3.2, the Transfer Entropy with a Kernel Estimator does not find the causal relations, where the $CLOUDCOVER-BASE$ is the cause. Overall, the result set of the Transfer Entropy with a Kernel Estimator has more results than the result set of the Granger Causality, but also seems to have a higher number of false positive results.

6.4.3 Experiment 4: Influence of Dataset Characteristics

In this section of the thesis, we describe an experiment, which tests the influence of changing dataset characteristics on the outcome of the Transfer Entropy algorithm with an Kernel Estimator.

Hypothesis

To conduct this experiment we create three *alternative hypotheses*, which are evaluated in the following:

h_{an}: The number of records/property in a dataset influences the result of the Transfer Entropy with a Kernel Estimator and an increase or decrease of the number of records/property changes the result.

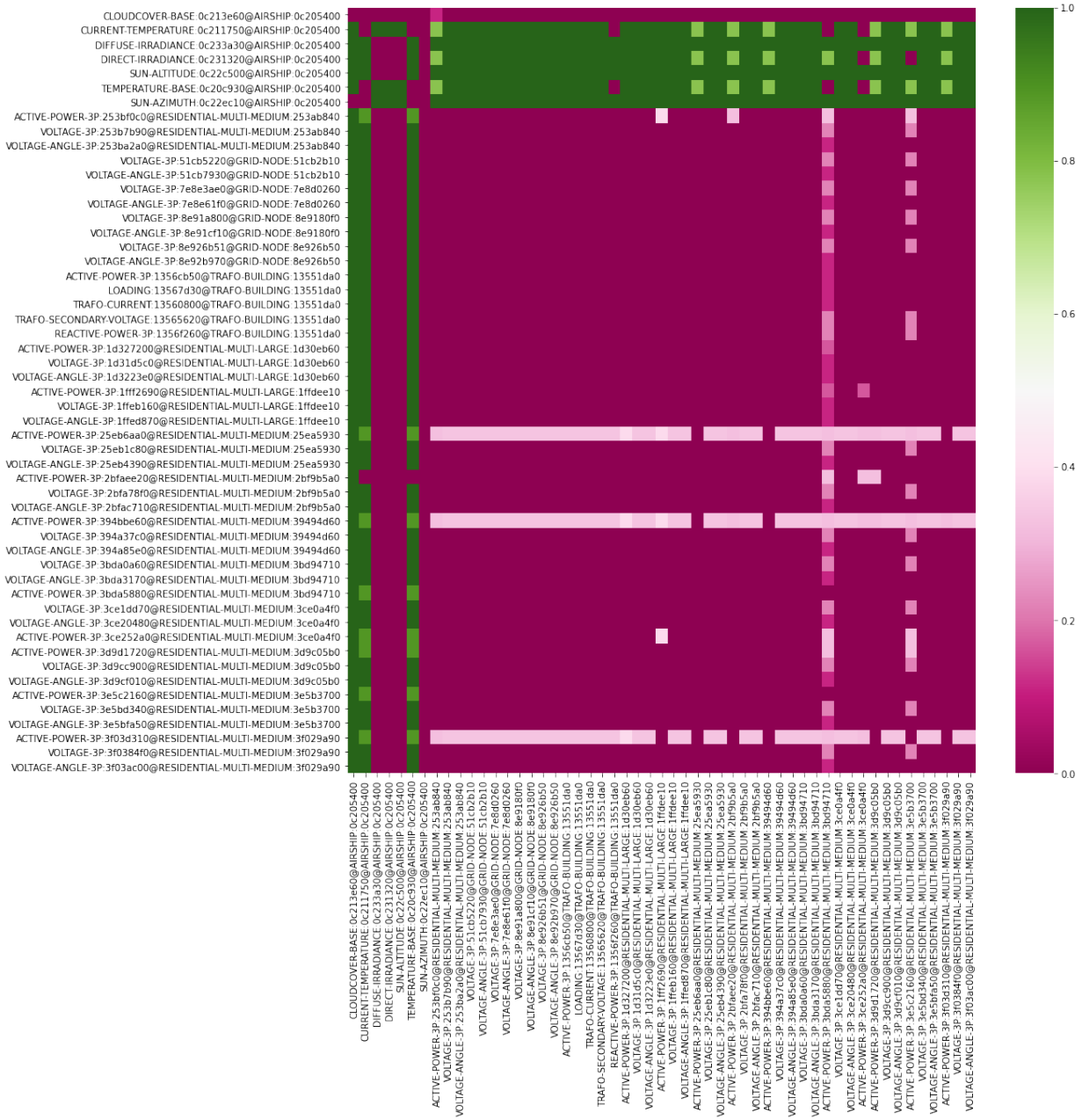


Figure 6.14: Transfer Entropy with a Kernel Estimator: Matrix representation of R_s of the Certainty by object information. Legend: The color scale indicates the C_o from violet (0%) to green (100%)

Cause		Effect		Certainty by object information
Feature of Interest	Property	Feature of Interest	Property	
AIRSHIP	DIFFUSE-IRRADIANCE	AIRSHIP	CLOUDCOVER-BASE	100%
RESIDENTIAL-MULTI-MEDIUM	VOLTAGE-ANGLE-3P	RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	11%
AIRSHIP	CURRENT-TEMPERATURE	RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	100%
RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	TRAFO-BUILDING	ACTIVE-POWER-3P	34%
RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	TRAFO-BUILDING	TRAFO-SECONDARY-VOLTAGE	34%

Table 6.8: Sample of R_s of the Transfer Entropy with a Kernel Estimator

h_{ac}: The Feature Type complexity FTc of a dataset influences the result of the Transfer Entropy with a Kernel Estimator and therefore a changing FTc changes the result.

h_{af}: The Measurement frequency Mf of a dataset influences the result of the Transfer Entropy with a Kernel Estimator and therefore a changing Mf changes the result.

Context

Similar to previous experiments, this experiment is conducted in the BIFROST simulation engine. For each dataset used in the experiment, a new testrun is started. Therefore, each testrun can be seen as *subject* of the experiment. The parameterization of the algorithm stays the same during the whole experiment. We further aim to evaluate the results of the algorithm, which therefore is the *object* of the experiment.

In Section 6.5.2 we already showed the optimal parameterization ($b = 0, 4, hl = 2$) for the algorithm which we will use in this experiment. Further, we set $cv = 0, 13$, which is the average of the discovered, optimal range.

Procedure

The experiment has multiple testruns which have to be conducted after each other. Therefore, the evaluator starts with the average dataset (Dataset 4) in the first testrun. Afterwards, the dataset is changed and the next testrun is executed. For each hypothesis different datasets are used. For h_{af} datasets 17-20 are used (Mf varies from 1 to 120

min), for h_{an} datasets 2, 4, 7 and 9 (number of datapoints/property varies from 10 to 500) and for h_{ac} datasets 1, 4, 8 and 10 (FTc varies from 2 to 9 Feature Types) are used. These datasets always have one parameter changed from Dataset 4. After finishing a testrun the evaluator conducts a manual inspection to find possible false positive results. He further documents the result and the used dataset.

Variables

The experiment has three *independent variables*, which are the three characteristics of the dataset. Therefore, the independent variables are:

- the Feature Type complexity FTc
- the Measurement frequency Mf
- the no. of datapoints/property

The *dependent variable* is the performance, which is represented by the *runtime* of the algorithm, the *Certainty by object information (C_o)* and the *number of results*.

Evaluation

In the evaluation we have to accept or reject all three hypotheses. If we accept the alternative hypothesis, we can automatically reject the null hypothesis and the other way around.

The first hypothesis we evaluate is h_{an}, which is the influence of the **number of records/property** on the result of the algorithm. In Figure 6.15 and 6.16 we see C_o and the no. of results over an increasing number of records/property.

We can see a stable C_o and a slightly decreasing number of results. The outlier at 10 datapoints/properties showed in the manual inspection almost exclusively false positive results. Therefore we can follow, a high number of records/properties reduces the number of true and false positive results, because both C_o and the no. of records decrease. We can also accept h_{an} and therefore reject the null hypothesis h_{0n}.

In the next step, we evaluate hypothesis h_{ac}, which focuses on the change of C_o for an increasing **Feature Type complexity** FTc. As the number of total Features of Interest increases in a more complex settlement, we cannot compare the number of results for this hypothesis.

In Figure 6.17 we see C_o over an increasing FTc. We see an initial drop in C_o, but a stable evolution afterwards. Therefore, C_o gets worse with an increasing FTc but does not get below a certain threshold (60%). We can still clearly see an influence of FTc on the algorithm and therefore accept h_{ac} and reject the corresponding null hypothesis h_{0c}.

The last remaining hypothesis is h_{af}, where the **Measurement frequency** Mf is changing. In Table 6.9 we can see the collected data to evaluate h_{af}. We see, that

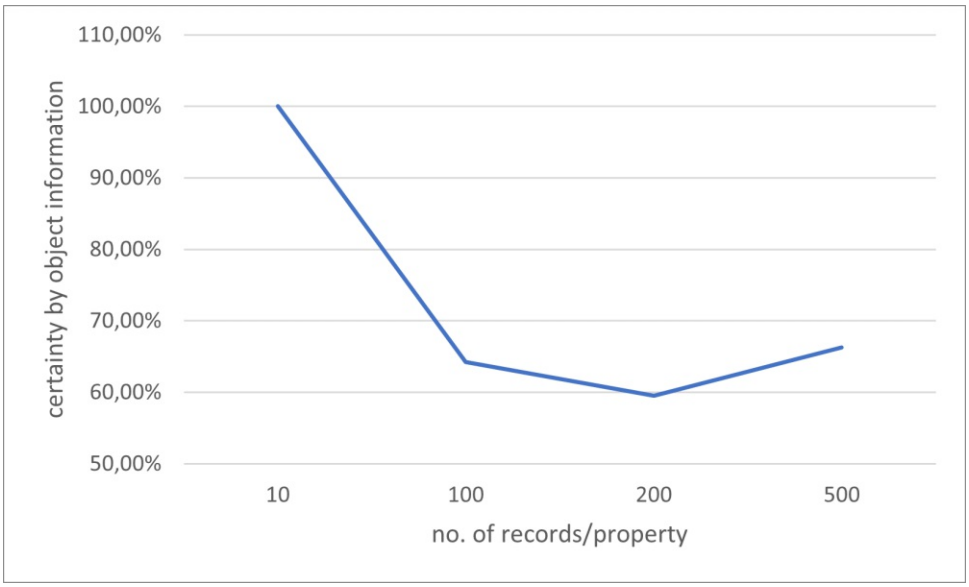


Figure 6.15: Transfer Entropy with a Kernel Estimator: Evaluation of records/property (C_o)

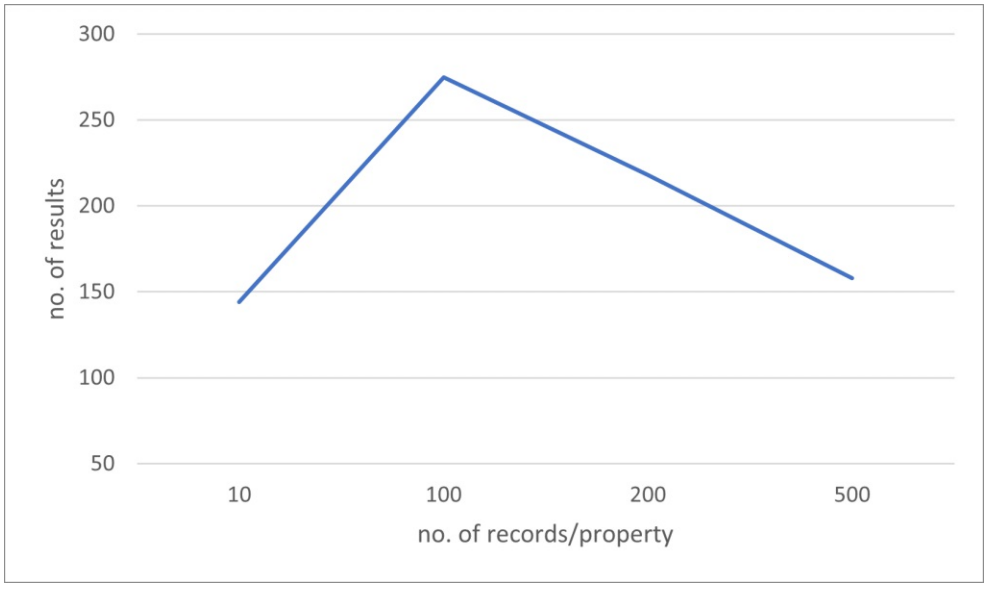


Figure 6.16: Transfer Entropy with a Kernel Estimator: Evaluation of records/property (number of results)

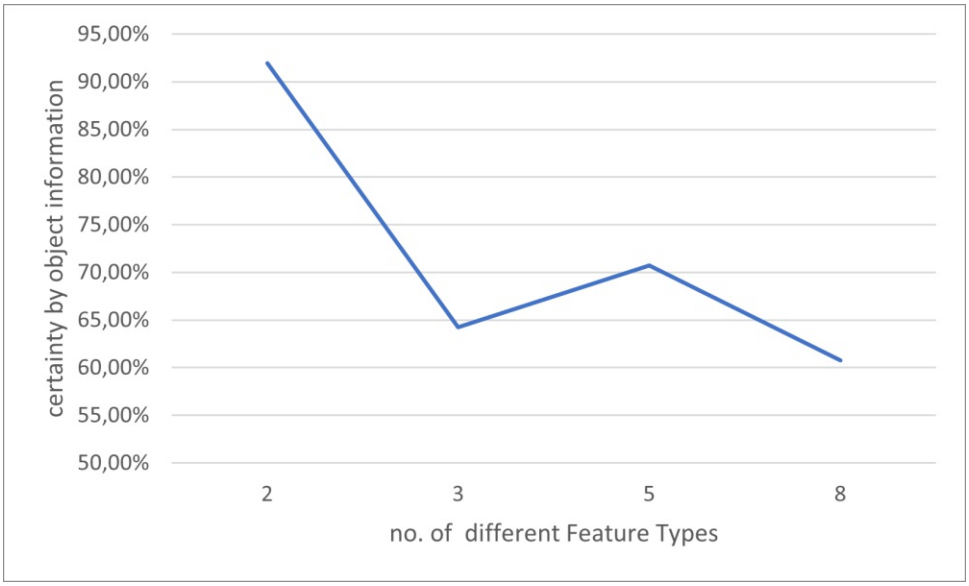


Figure 6.17: Transfer Entropy with a Kernel Estimator: Evaluation of FTc (C_o)

Dataset	**Measurement frequency**	**Certainty by object information**	**no. of results**
18	1 min	-	0
18	15 min	64,25%	275
18	60 min	85,60%	822
18	120 min	89,25%	893

Table 6.9: Transfer Entropy with a Kernel Estimator: Evaluation of Mf

$Mf < 1$ minute is not optimal, as the algorithm does not find a causal relation. Further $Mf > 120$ minutes is also not optimal, as the algorithm skips whole spikes in a timeseries and therefore finds false positive causal relations.

The remaining frequencies are between 15 and 60 minutes, where the testrun with a $Mf = 60$ minutes yields a high number of false positive events and therefore performs worse than the $Mf = 15$ minutes.

Therefore we can summarize: The only Measurement frequency useable in the BIFROST simulation engine without a high amount of false positive results is $Mf = 15$ minutes. Therefore we can accept h_{af} and reject h_{0f}.

Conclusions. After performing the experiment, we can come to the following conclusions:

- The higher the number of properties, the lower the number of results, while C_o stays the same.
- The higher FTc, the lower C_o, but C_o does not drop below a threshold.

- The only useable frequency is $Mf = 15$ minutes.

6.5 Transfer Entropy with a Kraskov Estimator

The Transfer Entropy with the Kraskov Estimator uses the algorithm described in Section 4.2.2 to acquire a set of causal relations. The library used to implement the Transfer Entropy with a Kraskov Estimator is provided by the github repository in [29].

6.5.1 Parameters

There are multiple different parameters to optimize the results of the Transfer Entropy with a Kraskov Estimator. In the parameters we distinguish two types of parameter sets:

- **Manual parameterization**: For this set of parameters we set three different parameters. These parameters are the **historic length** hl, the **critical value** cv and the **number of nearest Neighbors** $nonN$ used. hl is the same as described in Section 6.4.1. Further cv is the threshold which has to be surpassed for a value to be accepted. The difference to the Kernel Estimator is the unit, where the Kernel Estimator uses nats, while the Kraskov Estimator uses bits. The last parameter used for the manual parameterization is the number of neighbors used in the Nearest Neighbors algorithm, described in Section 4.2.2

- **Automatic Parameterization**: The algorithm provides an option to use the Ragwitz criterion [RK02] to estimate hl and $nonN$. Therefore the algorithm estimates both parameters for each pair of causal relations tested. The only set parameter is the **critical value**.

6.5.2 Experiment 5: Optimal Parameterization

This section describes the influence of the parameters on the Transfer Entropy with a Kraskov Estimator. There is a special focus on the use of the automatic estimation parameter.

Hypothesis

In this experiment we follow the controlled experiment method from Wohlin, presented in [WRH+12]. Therefore, we state the following *alternative hypothesis*:

h_m: The critical value cv, the historic length hl and the number of nearby neighbors $nonN$ parameters of the Transfer Entropy with a Kraskov Density Estimator influence the result of the algorithm and therefore an optimal set of these parameters can be defined.

h_a: The critical value cv of the Transfer Entropy with a Kraskov Estimator influences the result of the algorithm while using an automatic estimation method for the historic

Hypothesis	Independent variable	Scale
h_a	historic length	1 - 5
h_a	number of nearby neighbors	1 - 5
h_a	critical Value	0.02 - 0.1
h_a	enable automatic parameterization	true/false

Table 6.10: Independent Variables in the experiment for the parameterization of the Transfer Entropy with a Kraskov Estimator

length *hl* and the number of nearby neighbors *nonN* parameters and therefore an optimal *cv* can be defined.

h_am: The manual setting of parameters outperforms the automatically estimated setting and therefore the manual setting, with the optimal used parameters result in a better performance.

Context

To accept or reject the hypothesis, the evaluator conducts multiple test runs in the BIFROST simulation engine. Therefore, each test run is a *subject* of the experiment. We further evaluate the results of the algorithm, which therefore is the *object* of the experiment. Similar to previous experiments, we use an average dataset, namely the average dataset Dataset 4, as we only evaluate the parameters of the algorithm.

Procedure

The Evaluator, executing the experiment, starts the first test run with the lowest possible settings for each parameter and afterwards changes parameters for each consecutive test run. After finishing a test run, the evaluator has to check manually the results and document the results and the current setting of the parameters.

Variables

For this experiment, we define four *independent variables* which are the parameters of the algorithm:

- historic length *hl*
- number of nearby neighbors *nonN*
- critical Value *cv*
- enable automatic parameterization

An overview over the independent variables is shown in Table 6.10. The *dependent variable* is the performance, which takes the *runtime*, the *Certainty by object information* (C_o) and the *no. of results* into account.

Evaluation

In this evaluation we want to accept or reject all three hypotheses stated for this experiment. We therefore have to take a look at the results of the experiment and find the influence of the parameter on the algorithm.

We start by evaluating h_m, where we have to proof the influence of the **manual parameterization** on the result set of the causality acquisition algorithm. In Table 6.11 we can see the metrics for each combination of hl and $nonN$ while keeping cv stable. We show these two parameters in combination to show the volatility in the result set while changing these parameters. In Table 6.11 we can see the following statements are true:

- All three rows with more than 1000 results show a high number of false positive results.
- All six rows with $C_o > 90\%$ have a high number of false positive results.
- The most results are provided $nonN = 1$, while hl does not influence the result.
- The highest C_o is recorded with $hl = 3$, while $nonN$ does not influence the result.

Therefore, for an average dataset, the optimal parameterization is $hl = 3$ and $nonN = 1$. This combination of parameters seems to be optimal for this dataset, but as the difference in performance is high and unstable, the results also point towards the use of an automatic estimation of the parameters to optimize the results. We further accept the alternative hypothesis h_m.

After evaluating the manual parameterization, we take a look at the **automatic parameterization** and therefore, the only parameter, which has to be chosen, is cv. In Figure 6.18 we can see C_o over an increasing cv. We see a stable C_o at the start, which decreases the higher cv gets. This can be explained by the number of true positive results, which are removed from R_s. Therefore, we can accept $cv = 0,02$ as optimal critical value, as it has the highest number of results and a high C_o. We can therefore accept the alternative hypothesis h_a.

In the last step we want to **compare the automatic and the manual parameterization**, where we compare them based on the three performance criteria (number of results, C_o, runtime).

Both, the manual and the automatic parameterization yield around the same number of results, therefore they are equal in this metric. In terms of C_o the automatic parameterization yields a better result as the C_o is about 10% higher.

The last metric is the runtime where manual parameterization outperforms the automatic one, as the automatic parameterization needs to estimate the parameters for each combination of properties. For Dataset 4 the runtime, independent of the parameterization, was around 120 seconds. In the manual parameterized setting, each test run took below 10 seconds.

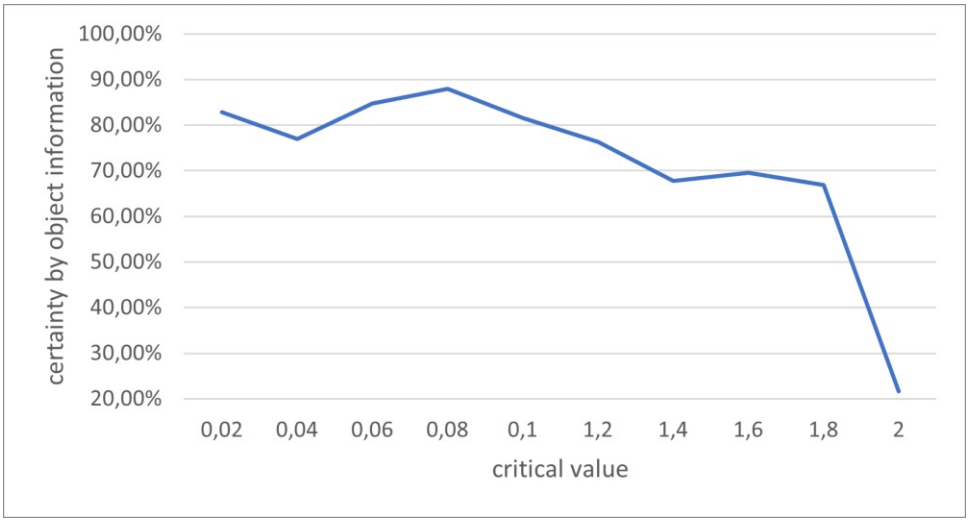


Figure 6.18: Transfer Entropy with a Kraskov Estimator: Evaluation of automatic parameterization

Conclusions. We conclude this section with the following findings: The manual parameterization has the better runtime, while the automatic parameterization has a better C_o. We can also accept $h_a m$.

Examples of Identified Causality Relations (R_s)

To provide an overview over results of the Transfer Entropy with a Kraskov Estimator, we show the following presentations of the result:

- Table 6.12 shows 6 of the 613 found by the causality acquisition algorithm, with the automatic parameterization and $cv = 0,13$. For example the table shows a causal relation between the Property $ACTIVE-POWER-3P$ of the Feature of Interest $TRAFO-BUILDING$ (cause) and the Property $ACTIVE-POWER-3P$ of the Feature of Interest $RESIDENTIAL-MULTI-LARGE$ (effect). This causal relation was identiefed by the CAA between every $TRAFO-BUILDING$ and every $RESIDENTIAL-MULTI-LARGE$, as $C_o = 100\%$.
- Similar to previous sections the results of the Transfer Entropy with a Kraskov Estimator are shown in Figure 6.19. This figure, similar to previous results, shows a lot of causal relations between Properties related to the weather (FI: AIRSHIP). Further we can see the influence of buildings on the Feature Type $RESIDENTIAL-MULTI-LARGE$, which presents an apartment house in a Settlement. The Transfer Entropy with a Kraskov Estimator is the only algorithm, which is able to find causal relations between Features of Interest independent of the weather. The CAA also finds some causal relations between Features of Interest with a lower C_o.

Historic Length	k nearest Neighbor	Certainty by object information	no. of results
1	1	76.69%	686
1	2	90.72%	1237
1	3	94.3%	1138
1	4	78.42%	586
1	5	69.59%	316
2	1	75.22%	684
2	2	92.25%	1115
2	3	66.95%	343
2	4	75.65%	124
2	5	76.43%	82
3	1	76.55%	687
3	2	75.42%	538
3	3	83.23%	176
3	4	60.59%	52
3	5	65.81%	68
4	1	76.62%	680
4	2	74.23%	343
4	3	83.36%	106
4	4	91.61%	138
4	5	90.33%	136
5	1	74.52%	692
5	2	73.69%	173
5	3	87.8%	84
5	4	43.43%	15
5	5	45.42%	21

Table 6.11: Transfer Entropy with a Kraskov Estimator: Evaluation of the historic length and k nearest neighbor parameters

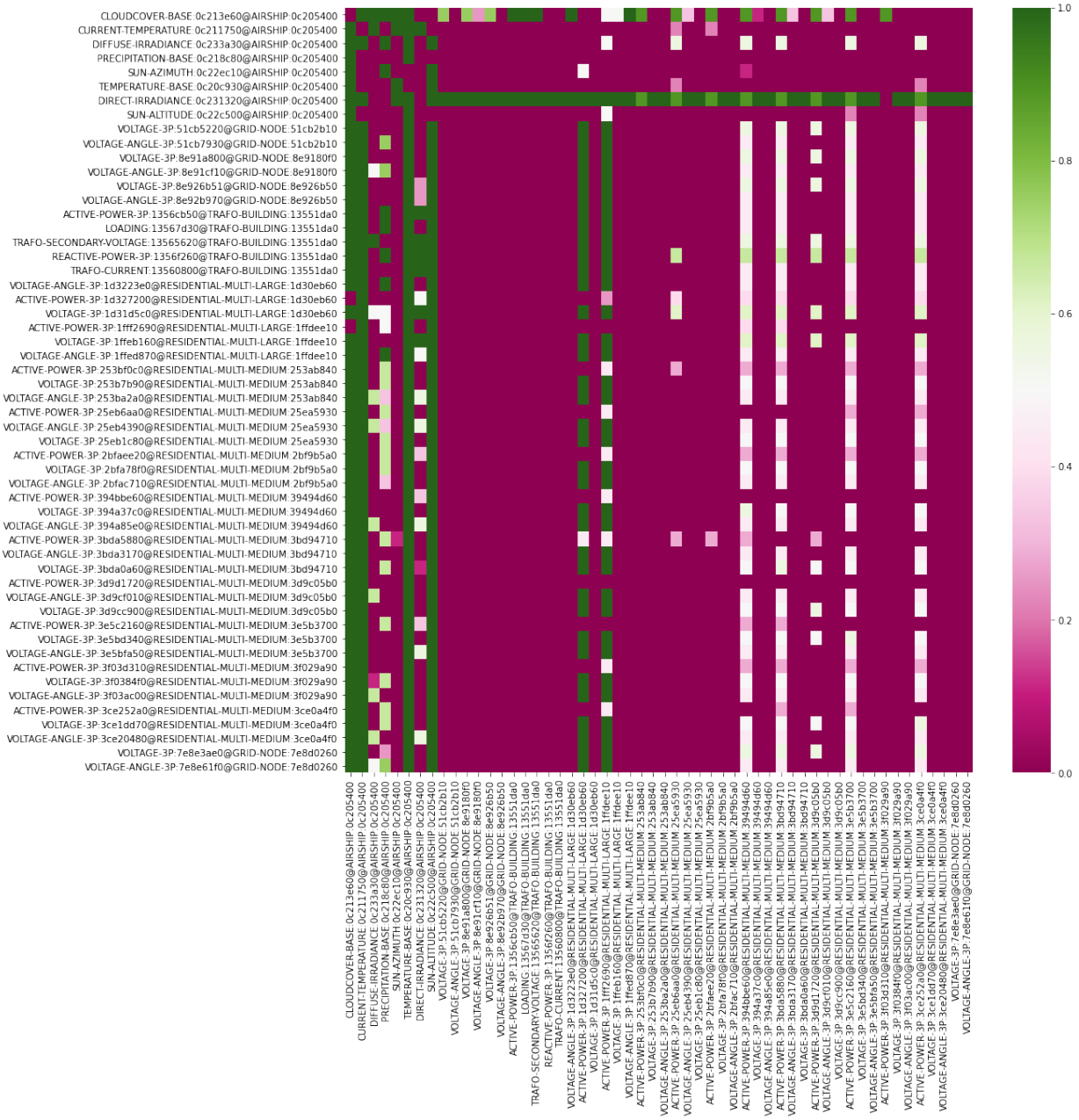


Figure 6.19: Transfer Entropy with a Kraskov Estimator: Matrix representation of R_s of the Certainty by object information. Legend: The color scale indicates the C_o from violet (0%) to green (100%)

Cause		Effect		Certainty by object information
Feature of Interest	Property	Feature of Interest	Property	
TRAFO-BUILDING	ACTIVE-POWER-3P	RESIDENTIAL-MULTI-LARGE	ACTIVE-POWER-3P	100%
TRAFO-BUILDING	TRAFO-CURRENT	RESIDENTIAL-MULTI-LARGE	ACTIVE-POWER-3P	100%
TRAFO-BUILDING	REACTIVE-POWER-3P	RESIDENTIAL-MULTI-LARGE	ACTIVE-POWER-3P	67%
RESIDENTIAL-MULTI-MEDIUM	VOLTAGE-3P	RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	56%
RESIDENTIAL-MULTI-MEDIUM	ACTIVE-POWER-3P	RESIDENTIAL-MULTI-LARGE	ACTIVE-POWER-3P	34%
AIRSHIP	DIFFUSE-IRRADIANCE	AIRSHIP	SUN-ALTITUDE	100%

Table 6.12: Sample of R_s of the Transfer Entropy with a Kraskov Estimator

6.5.3 Experiment 6: Influence of Dataset Characteristics

In this experiment we observe the behavior of the Transfer Entropy with a Kraskov Estimator during changing dataset characteristics.

Hypothesis

For this experiment we evaluate the following three hypotheses:

h_{an}: The number of records/property in a dataset influences the result of the Transfer Entropy with a Kraskov Estimator and an increase or decrease in the number of records/property changes the result.

h_{ac}: The Feature Type complexity FTc of a dataset influences the result of the Transfer Entropy with a Kraskov Estimator and therefore a changing FTc changes the result.

h_{af}: The Measurement frequency Mf of a dataset influences the result of the Transfer Entropy with a Kraskov Estimator and therefore a changing Mf changes the result.

Context

The experiment is carried out in the BIFROST simulation engine, where for each used

dataset a test run is started, while the parameterization of the algorithm stays the same. Similar to previous experiments, each test run is a *subject* of the experiment, while the algorithm is the *object* of the experiment. We also use the optimal parameterization (automatic parameterization, $cv = 0,02$) of Section 6.5.2 to evaluate the characteristics of the dataset.

Procedure

During the execution of the experiment, the evaluator starts a test run with an average dataset (Dataset 4) and consecutively starts test runs with datasets which have one characteristic changed from the starting dataset. For h_{af} datasets 3-6 are used (Mf varies from 1 to 120 min), for h_{an} datasets 2, 4, 7 and 9 (number of datapoints/property varies from 10 to 500) and for h_{ac} datasets 1, 4, 8 and 10 (FTc varies from 2 to 9 Feature Types) are used. These datasets always have one characteristic changed from Dataset 4. After each test run the evaluator performs a manual inspection for possible false positive results and documents the results and the used dataset.

Variables

This experiment contains three *independent variables*. These variables are the three characteristics of a dataset:

- the Measurement frequency Mf
- the Feature Type complexity FTc
- the no. of datapoints/property

The *dependent variable* of the experiment is the performance, which is conducted from the *runtime*, the *Certainty by object information (C_o)* and the *no. of results*.

Evaluation

In this evaluation we either want to accept or reject all three hypotheses. We therefore evaluate them one after the other. For each hypothesis we take a look at the documented results of the experiment and show the influence the characteristic has on the result set R_s.

We start the evaluation by discussing hypothesis h_{af}, the influence of the **Measurement frequency** Mf on the results of the algorithm. In Table 6.13 we can see the results of the experiment, where we can see either a high number of results ($Mf = 1min, 60min$), where R_s contains a high number of false positive results or a low C_o ($Mf = 120min$). The only remaining value for Mf, available in BIFROST, is 15 minutes, which seems to be the only valid setting.

After showing the results of Mf, we take a look at hypothesis h_an, where we evaluate the **number of records/datapoint**. In Table 6.14 we can see the results of the experiment. In this table we see an increase in runtime, while increasing the number of

Dataset	Mf	Certainty by object information	no. of results
17	1 min	74,16%	1686
18	15 min	82,87%	613
19	60 min	80,67%	2536
20	120 min	61,65%	775

Table 6.13: Transfer Entropy with a Kraskov Estimator: Evaluation of the Mf

Dataset	no. of datapoints/ property	Certainty by object information	no. of results	Runtime
14	10	-	0	-
18	100	82,87%	613	128s
22	200	72,95%	780	191s
37	500	88,97%	470	394s

Table 6.14: Transfer Entropy with a Kraskov Estimator: Evaluation of the no. datapoints/property

records/datapoint. Further we can see the inability of the algorithm to acquire a result with 10 datapoints. We can also see an increasing number of datapoints/property increases C_o, while decreasing the number of results, the higher the number of records/datapoint gets. At 200 datapoints, we see an outlier, where the algorithm finds a high number of causal relations and has a lower C_o due to a higher number of false positive results for Dataset 7. We can therefore see, the higher the number of records/datapoints gets, the higher the runtime, the higher C_o and the lower the number of results gets. We can also reject the null hypothesis h_0n and accept the alternative hypothesis h_an.

The last evaluated hypothesis is h_ac, where we take a look at the **Feature Type complexity** FTc. We can only evaluate C_o, as the runtime will increase the more Features of Interest a Settlement contains and so will the number of results. In Figure 6.20 we can see C_o over an increasing FTc. We can see a decline of C_o, while increasing FTc, even though the decline in C_o is lower than in previous algorithms.

Conclusions. In the end, we can conclude the following statements for the Transfer Entropy with a Kraskov Estimator:

- $Mf = 15$ minutes seems to be the only possible setting.
- The higher the number of records/datapoints gets, the higher the runtime, the higher C_o, the lower the number of results gets.

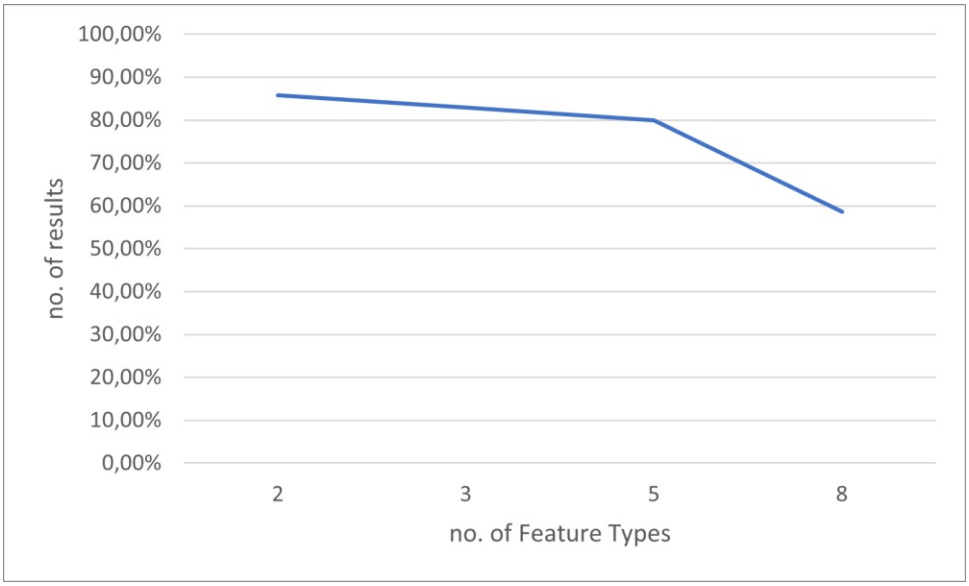


Figure 6.20: Transfer Entropy with a Kraskov Estimator: Evaluation of FTc (Certainty by object information)

- The higher FTc gets, the lower C_o gets.

6.6 Peter-Clark Algorithm

The Peter-Clark Algorithm (PC Algorithm) tries to build a DAG to represent found causal relations of a set of timeseries, as described in Section 4.2.3. The implementation uses the library provided by [30]. As previously mentioned, the PC-algorithm can only solve P2.

6.6.1 Parameters

The PC-algorithm has two different parameters to optimize R_s. These parameters are the **depth** d and **alpha** a. Alpha is the alpha level for the independence test used in the algorithm, while the depth limits the maximal depth of the algorithm. Therefore, a depth of five limits the algorithm to test for independence between two properties under the condition of at most five other properties.

6.6.2 Experiment 7: Optimal Parameterization

This section shows an experiment, which aims to find the influence of the parameterization on the result. Further we aim to find an optimal setting for the parameters.

Hypothesis

For this experiment we use the following hypothesis:

Hypothesis	Independent variable	Scale
h_a	depth	1 - 5
h_a	alpha	0.05 - 0.35

Table 6.15: Independent variables in the experiment for the parameterization of the PC-algorithm

h_m: The parameters of the PC-algorithm influence the result of the algorithm and therefore an optimal set of these parameters can be defined.

Context

The experiment takes place in the context of the BIFROST simulation engine. For each possible set of parameters, a new test run is started. By the definition of a subject, each test run is a *subject* of the experiment, while the algorithm is the *object* of the experiment, as we aim to evaluate the results of the algorithm. As we want to evaluate the parameters of the algorithm and not the characteristics of the dataset, we use an average dataset, namely the average dataset Dataset 4 for the experiment.

Procedure

During the execution of the experiment the evaluator performs successive test runs for each possible combination of parameters. Therefore the evaluator starts with the lowest possible combination of parameters and changes the parameters after each test run. The experiment is finished after each possible combination has been used. After each test run the evaluator performs a manual inspection of R_s and documents the results of the test run.

Variables

The experiment has two *independent variables*, which are the parameters of the algorithm. Therefore, the two *independent variables* are:

- alpha a
- depth d

An overview is shown in Table 6.15. Further the experiment has the *dependent variable* performance, which is conducted from the *runtime*, the *Certainty by object information* (C_o) and the *number of results*.

Evaluation

Similar to previous evaluations we want to find the optimal parameters to optimize the performance. Therefore, we want to maximize C_o and the number of results while keeping the runtime as low as possible.

In Table 6.16 the evaluation results of the PC-algorithm are shown. We can see that an increase of d exponentially increases the runtime. While $d = 1$ takes under a second, $d = 5$ takes around half an hour and $d = 6$ takes around five hours.

With the depth reachable in a justifiable time, the algorithm is not able to provide a result set with a reasonable collection of causal relation. We can see, independent of d, the result set stays the same and contains a high number of false positive results.

We can further see no change in C_o, during a change in either parameter, as this metric stays between 50 and 60%. The only change, we were able to observe is a decrease of found results while increasing d or a.

Due to the inability of the algorithm to deliver a result set without a high number of false positive results in a reasonable time, we can follow the PC-algorithm should not be applied to data from the BIFROST simulation engine.

6.7 Comparison of Results

In this section we compare the results of the algorithms based on the Certainty by ensemble based learning. For the comparison we use the average dataset, namely Dataset 4. Further each algorithm is parameterized using the optimal parameterization, found in Experiment 1, 3 and 5. A summary of these parameters can be seen in Section 6.8.1. As the Transfer Entropy with a Kraskov Estimator has the highest C_o, we compare the other two algorithms to the Transfer Entropy with a Kraskov Estimator. In the following Figures the algorithms are compared:

- In Figure 6.21 we can see a matrix representation of the Certainty by ensemble based learning of the Transfer Entropy with a Kraskov Estimator and the Granger Causality. C_e is the percentage of causality acquisition algorithms, which found the causal relation. Therefore, if both the Transfer Entropy with a Kraskov Estimator and the Granger Causality find a causal relation $C_e = 100\%$, if only one of the algorithms finds the relation $C_e = 50\%$. In Figure 6.21 $C_e = 100\%$ each unique pair of Features of Interests and Properties is shown on both axis. The y-axis presents causes, while the x-axis presents effects. A green square represents $C_e = 100\%$, while a white square represents $C_e = 50\%$ and a violet square presents $C_e = 0\%$. Therefore the causal relation between the $CLOUDCOVER - BASE$ and the $VOLTAGE - ANGLE - 3P$ in the top right corner (white square) has a $C_e = 50\%$, while the green square next to the white square in the top right corner presents a causal relation, with $C_e = 100\%$ between the $CLOUDCOVER - BASED$ and $VOLTAGE - \S P$.
- In Figure 6.22 we can see a matrix representation of the Certainty by ensemble based learning of the Transfer Entropy with a Kraskov Estimator and the Transfer Entropy with a Kernel Estimator. This figure has the same color codes as Figure 6.21.

Depth	Alpha	Certainty by object information	no. of results	Runtime
1	0.05	52.63%	225	0s
1	0.1	56.25%	251	0s
1	0.15	58.07%	271	0s
1	0.2	55.69%	281	0s
1	0.25	52.54%	302	0s
1	0.3	54.72%	324	0s
1	0.35	58.88%	342	0s
2	0.05	56.79%	96	1s
2	0.1	54.67%	107	1s
2	0.15	57.25%	111	1s
2	0.2	55.71%	104	1s
2	0.25	58.71%	103	1s
2	0.3	58.15%	112	1s
2	0.35	60.09%	117	1s
3	0.05	56.79%	96	10s
3	0.1	54.67%	107	10s
3	0.15	57.25%	111	10s
3	0.2	55.71%	104	10s
3	0.25	58.71%	103	10s
3	0.3	58.03%	110	10s
3	0.35	58.56%	114	10s
4	0.05	56.79%	96	149s
4	0.1	54.67%	107	150s
4	0.15	57.25%	111	148s
4	0.2	55.71%	104	150s
4	0.25	58.65%	102	148s
4	0.3	58.03%	110	152s
4	0.35	58.56%	114	151s
5	0.05	56.79%	96	1836s
5	0.1	54.67%	107	2107s
5	0.15	57.25%	111	1945s
5	0.2	55.71%	104	1888s
5	0.25	58.65%	102	1866s
5	0.3	58.03%	110	1859s
5	0.35	58.56%	114	1818s

Table 6.16: PC-algorithm: Evaluation of the depth and alpha parameters

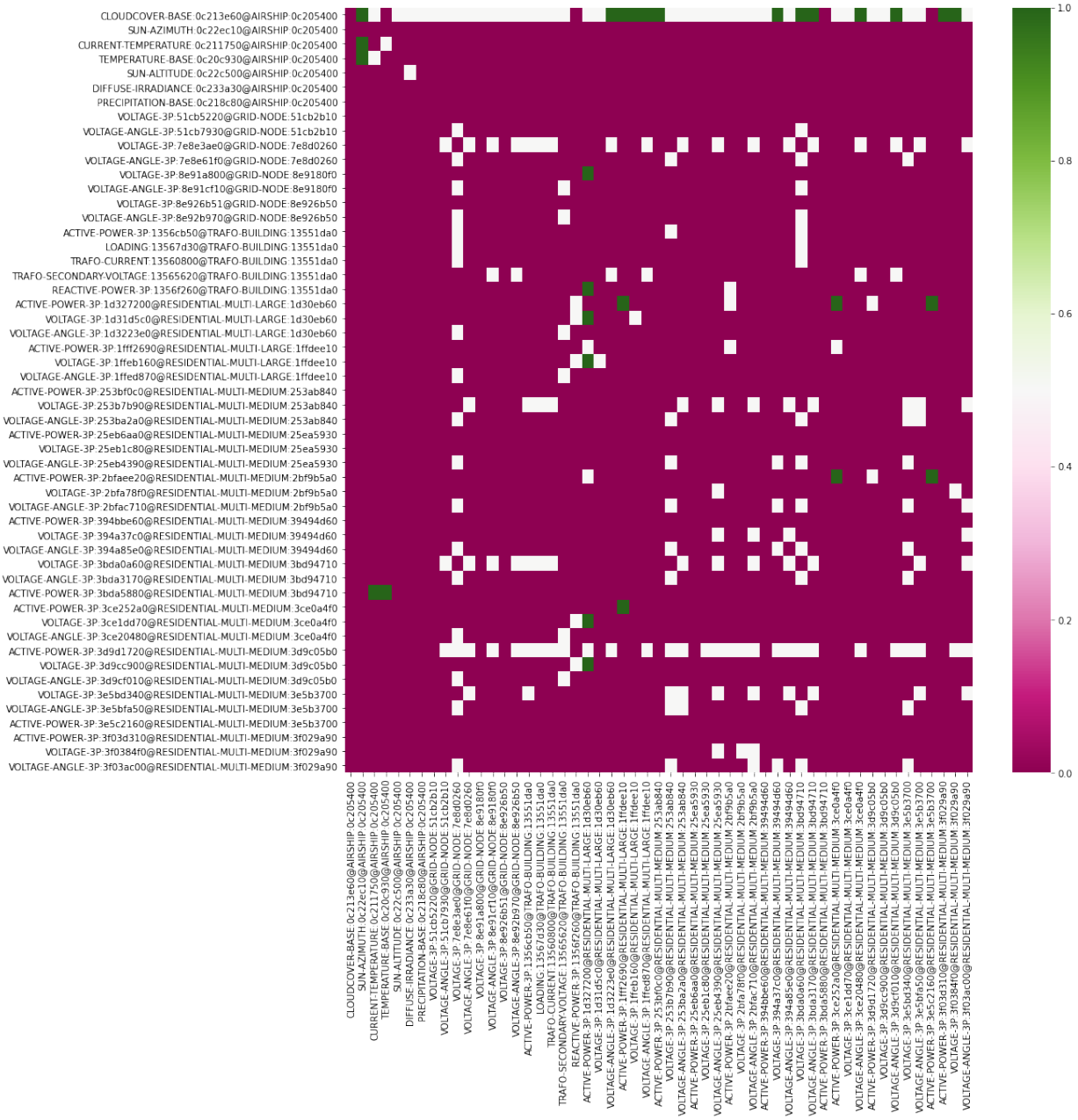


Figure 6.21: Transfer Entropy with a Kraskov Estimator compared to Granger Causality: Matrix representation of R_s of the Certainty by ensemble based learning. Legend: The color scale indicates the C_e from violet (0%) to green (100%)

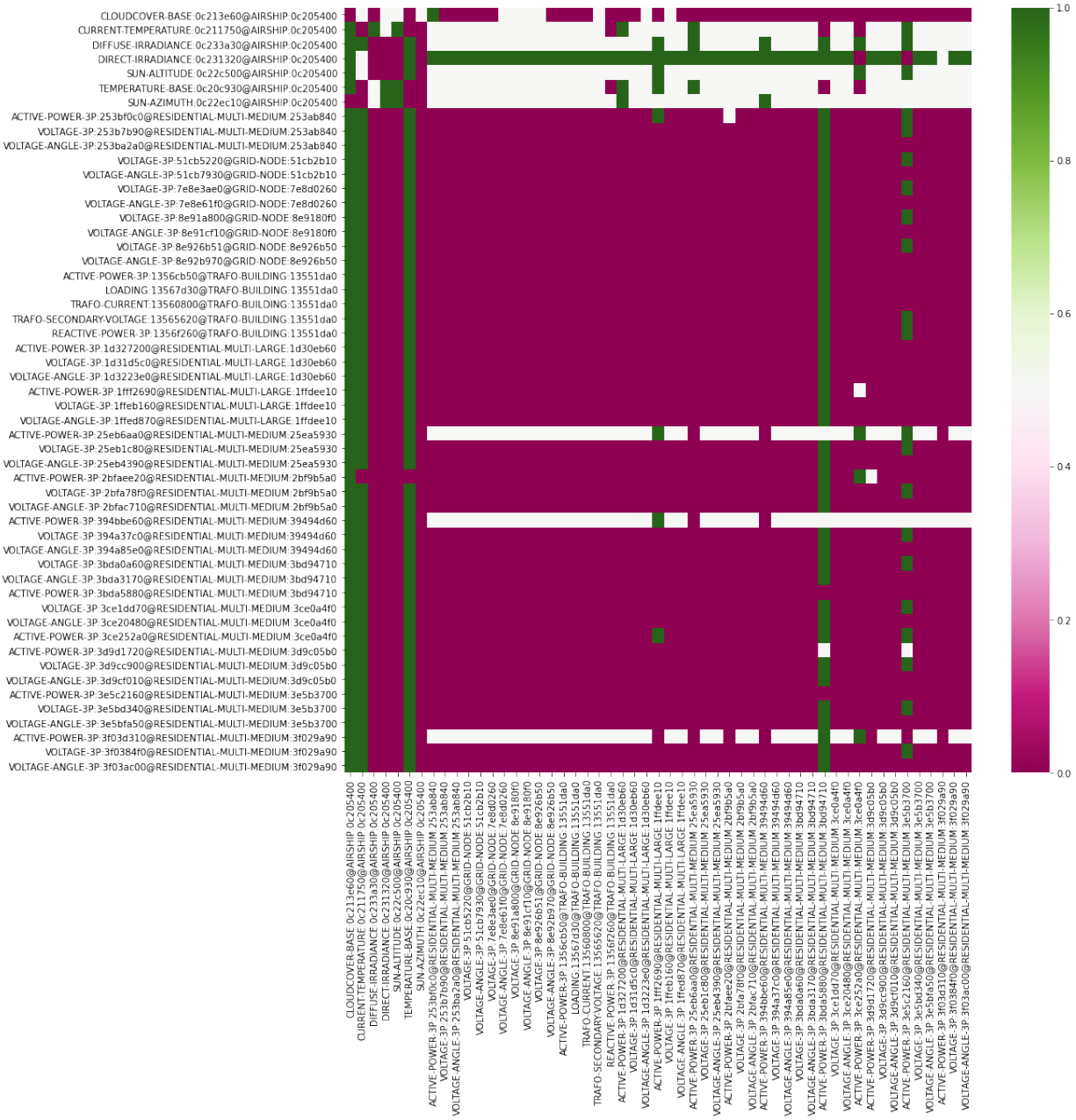


Figure 6.22: Transfer Entropy with a Kraskov Estimator & Transfer Entropy with a Kernel Estimator: Matrix representation of R_s of the Certainty by ensemble based learning. Legend: The color scale indicates the C_e from violet (0%) to green (100%)

In both comparisons we can see a set of Causal relations with $C_e = 100\%$ (green squares). For these relations we can assume they are true positives, because at least two causality acquisition algorithms found them. Further there are some relations, which are only found by one of the algorithms (white squares), where the manual inspection showed a higher number of false positive results in both the Transfer Entropy with a Kernel Estimator and the Granger Causality.

Especially the Transfer Entropy with a Kernel Estimator seems to be still biased towards certain Features of Interest. Therefore, even though the experiment provided a seemingly optimal parameterization, the Transfer Entropy outperforms the Transfer Entropy with a Kernel estimator using the automatic parameterization, regarding the amount of true positive results.

6.8 Evaluation Summary

In this chapter we investigated the influence of the algorithm parameters and the influence of the dataset characteristics on the result set R_s for each algorithm. We showed the applicability of the algorithms to BIFROST and their strengths and weaknesses. In this section we summarize the findings of the chapter.

6.8.1 Optimal Parameterization

After evaluating the influence of the parameterization, we found an optimal setting for three of the four implemented CAA. The PC-algorithm was not able to provide an acceptable result and has therefore no optimal parameterization in the context of the BIFROST simulation engine.

The optimal settings for the algorithms are:

- Granger Causality: $ls = 1$, $cv = 0,7$
- Transfer Entropy with a Kernel Estimator: $b = 0,4$, $hl = 2$, $cv = 0,1 - 0,15$
- Transfer Entropy with a Kraskov Estimator: use automatic parameterization, $cv = 0,02$

6.8.2 Influence of the dataset characteristics

We showed the influence of the dataset characteristics on the result set. Each of the three evaluated algorithms (Granger Causality, Transfer Entropy with a Kernel Estimator, Transfer Entropy with a Kraskov Estimator) behave similarly to changes in the dataset characteristics, as follows.

- **Measurement Frequency.** *All three algorithms operate the best at $Mf = 15$ minutes.* The only algorithm, which is also able to work with other frequencies

provided by BIFROST is the Granger algorithm, which delivers acceptable results until $Mf = 60$ minutes.

- **Feature Type complexity.** Each algorithm decreases in performance, when the FTc of the dataset increases. The best performing algorithm at a high FTc is the Transfer Entropy with a Kraskov Estimator.
- **Number of Datapoints** leads to quite different behavior across algorithms. While the Granger Causality performs strictly better under a higher number of datapoints, the Transfer Entropy with a Kernel Estimator keeps the performance the same as the Certainty by object information stays stable, while the number of results decreases. The last CAA, the Transfer Entropy with a Kraskov Estimator, has a higher accuracy the higher the number of results gets, as the Certainty by object information increases. The disadvantage of the higher accuracy is an increased runtime.

6.8.3 Conclusions and Recommendations

After evaluating each implemented CAA, we found one algorithm, which is not suitable for the BIFROST simulation engine (the PC algorithm), while the other three deliver results with a similar performance. Even though the performance is similar, the strengths and weaknesses are not, as the Granger Causality and the Transfer Entropy with a Kernel Estimator have a better runtime but a worse Certainty by object information than the Transfer Entropy. We therefore recommend the usage of the Transfer Entropy with a Kraskov Estimator, as long as the runtime is reasonable, to maximize the correctness of the found causal relations.

CHAPTER 7

Conclusion and Future Work

7.1 Conclusion

In this thesis we aim to solve problems related to causality and explainable Cyber Physical Systems (expCPS). In Section 1.3 we introduce four Research Questions that are investigated in this thesis. In the following we discuss these questions and summarize the corresponding answers/results based on the material presented in Chapters 2 – 6.

- **RQ1:** *What are classical ways to represent causality information and which formalisms and notions are used to represent causality information?*

  We investigate this question in Chapter 2, where we start by reviewing a collection of different definitions of causality and form a working definition of causality for expCPS based on these definitions. Afterwards we show different causality representations, used in a broad range of research areas. These representation methods show the different needs of systems in different application areas. Finally, we provide an overview of important aspects to consider when choosing or creating a causality representation method.

  The first aspect to consider is the difference between types of causal relations, where some systems require the representation to differentiate between causal relations. Further it can be important for a system to weight the relations, to distinguish the strength of relations. It can also be critical to consider multiple causes for the same event or object. Finally, we differentiated two types of causal relations: generic Causality and concrete Causality.

- **RQ2:** *Which ways are already used to acquire the information required to build a causality representation and how these approaches use the acquired information to build a causality representation?*

This Research Question focuses on different expCPS, which use a causality representation or a causality acquisition algorithm or both. In Chapter 3 we show, how these expCPS acquire causality knowledge and represent the gained knowledge. We also compare the expCPS, based on three aspects.

The first aspect we used, is the *application area*, where we found two systems, which already operate in a power grid. The first system [LÖC07] uses the Multilevel Flow Model to represent causality knowledge, while the second [MDEG18] one uses an advanced filter method to acquire events causally related to each other. The other two aspects are the *required knowledge of experts* to build and operate the system. Therefore, these aspects describe the required help of experts, the system needs to be initially built and to be used during daily operation.

Finally we use the results from Chapter 2 and 3 to propose the Multilevel Flow Model, as an optimal causality representation approach for an expCPS, as it provides the possibility to scale infinitely, because it represents functionality instead of the structure of components. It further can be adapted to include all important aspects identified in Section 2.4.

- **RQ3:** *What approaches are already used to acquire causality information from a system and is it possible to apply these approaches to an expCPS?*

  We explore RQ3 in Chapter 4 where we start by formally defining causality in expCPS in the problem of causality acquisition. In this part of the thesis, we formally describe each part related to a causal relation in the BIFROST simulation engine. Based on this problem definition, we discuss different algorithms to acquire causality knowledge from a system. We further show the difference in approaches to acquire knowledge. It is important to decide for the correct algorithm based on the system the algorithm is applied for. Therefore, we focus on three different aspects to consider, when choosing the correct algorithm. At the start it is important to consider both, *the input and the output data type* of the method. It is important to use an algorithm, which can handle the data provided by the system and provides data the system can use. The last considered aspect is the *possibility to use already collected knowledge*, as this knowledge can increase the performance of the algorithm.

  After reviewing different methods, we also discuss the applicability of the causality acquisition algorithms to the expCPS module built on top of the BIFROST simulation engine [AES+20]. We concluded that there are two approaches, which clearly fit to the module. These approaches are the Granger Causality [Gra69] and the Transfer Entropy (with different estimators) [Sch00]. We also found an algorithm, which can be applied to the BIFROST simulation engine, but has an exponentially worse runtime – the Peter Clark Algorithm [SGSH00].

- **RQ4:** *How can causality information, with the help of the approaches from RQ3, be acquired from the BIFROST simulation engine? What are advantages and disadvantages of each method?*

To acquire causality knowledge from the BIFROST simulation engine, we implemented a framework, which is able to apply the Granger Causality, two different implementations of the Transfer Entropy and the Peter Clark Algorithm to the expCPS module. Further the framework provides three different metrics, on which the evaluation of these algorithms is based. These metrics, presented in Chapter 5, are the **Certainty by Object information**, **the Certainty by ensemble based learning** and the **amount of causal relations**. In the framework we use the data acquired from the expCPS to provides a result set, with a collection of found causal relations and a list of metrics, which can determine the quality of the result.

Afterwards we use this data to conduct controlled experiments and evaluate the causality acquisition algorithms, based on the provided metrics. For each implemented algorithm, we conduct two experiments. In the first experiment for each algorithm, we aim to find the optimal parameterization of the algorithm. In the second experiment we investigate the influence of changing characteristics of the dataset. We conclude the experiments, with the observation, that the Transfer Entropy with a Kraskov Estimator provides the most optimal result ($C_o = 83\%$, number of results = 613 and runtime = 128s). The Transfer Entropy with a Kernel Estimator and the Granger Causality provide a good result on paper, but contain a high amount of false positive results. Finally, the Peter Clark Algorithm is unfeasible, as it is computationally too expensive and therefore the time-performance quickly deteriorates to unacceptable levels (several hours) even for small datasets.

7.2 Assumptions & Limitations

There are some limitations and assumptions in the thesis. The first limitation of the thesis is the focus on the BIFROST simulation engine. Therefore, we focused the research on a single expCPS and could not investigate the influence of the chosen causality representation and causality acquisition algorithms on different systems (or domains). Further the decision for the Multilevel Flow Model is only based on theoretical literature research and was not evaluated in either a simulated or real expCPS.

Further, there is currently no gold standard for causal relations in the BIFROST simulation engine. Therefore, the evaluations are only based on the metrics described in Section 5.3. There are also some risks, while performing the controlled experiments, described in [WRH+12]. One threat is the possibility of a low statistical sample size, as we only use four different settlements to evaluate the algorithms. Another threat is the expectation of the experimenter, where the experimenter expects a certain solution and manipulates the experiment towards a certain result. We tried to address this risk by frequent joint meetings with the supervision team in order to reduce bias from a single experimenter.

7.3 Future Work

During the thesis we were able to propose a causality representation, the Multilevel Flow Model, suitable for an explainable Cyber Physical System (expCPS). In the future, the Multilevel Flow Model should be implemented and evaluated in a simulated or real expCPS, to evaluate its usability. Further, the representation should be extended with more functionality such as by adding weights to the representation elements of the Multilevel Flow Model.

Another important research in the future is the implementation and evaluation of the algorithms shown in Chapter 4 in a real or another simulated environment to check, if the standards set in the thesis still apply for systems apart from the BIFROST simulation engine or other important aspects to consider in an expCPS are found. Therefore, a future work should investigate the performance of the causality acquisition algorithms and the applicability of causality representation in similar systems.

Finally, we envision combining both the representation and the acquisition of causality into a single module to investigate the interaction between these two components. Additionally, causal relations found by the algorithms should be evaluated quantitatively by domain experts.

List of Figures

List of Tables

Bibliography

[23] How to interpret the bandwidth value in a kernel density estimation? https://stats.stackexchange.com/questions/226232/how-to-interpret-the-bandwidth-value-in-a-kernel-density-estimation. Accessed: 2021-03-16.

[28] Grangercausality. https://github.com/mdrobek/GrangerCausality. Accessed: 2021-03-26.

[29] Java information dynamics toolkit (jidt). https://github.com/jlizier/jidt. Accessed: 2021-03-26.

[30] Tetrad. https://github.com/cmu-phil/tetrad. Accessed: 2021-03-26.

[AES+20] Peb R Aryan, Fajar J Ekaputra, Marta Sabou, Daniel Hauer, Ralf Mosshammer, Alfred Einfalt, Tomasz Miksa, and Andreas Rauber. Simulation support for explainable cyber-physical energy systems. In *2020 8th Workshop on Modeling and Simulation of Cyber-Physical Energy Systems*, pages 1–6. IEEE, 2020.

[BG11] Radhakisan Baheti and Helen Gill. Cyber-physical systems. *The impact of control technology*, 12(1):161–166, 2011.

[BS01] Luiz A Baccalá and Koichi Sameshima. Partial directed coherence: a new concept in neural structure determination. *Biological cybernetics*, 84(6):463–474, 2001.

[CI11] BD Craven and Sardar MN Islam. Ordinary least-squares regression. *The SAGE dictionary of quantitative management research*, pages 224–228, 2011.

[CSK+18] Jose Cordova, Lalitha Madhavi Konila Sriram, Ayberk Kocatepe, Yuxun Zhou, Eren E Ozguven, and Reza Arghandeh. Combined electricity and traffic short-term load forecasting using bundled causality engine. *IEEE Transactions on Intelligent Transportation Systems*, 20(9):3448–3458, 2018.

[DLC17] Jesse Dunietz, Lori Levin, and Jaime G Carbonell. The because corpus 2.0: Annotating causality and overlapping relations. In *Proceedings of the 11th Linguistic Annotation Workshop*, pages 95–104, 2017.

[DYCS13] Ping Duan, Fan Yang, Tongwen Chen, and Sirish L Shah. Direct causality detection via the transfer entropy approach. *IEEE transactions on control systems technology*, 21(6):2052–2066, 2013.

[GLV19] Joel Greenyer, Malte Lochau, and Thomas Vogel. Explainable software for cyber-physical systems (es4cps): Report from the gi dagstuhl seminar 19023, january 06-11 2019, schloss dagstuhl. *arXiv preprint arXiv:1904.11851*, 2019.

[Gra69] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[HS13] David Heckerman and Ross D Shachter. A definition and graphical representation for causality. *arXiv preprint arXiv:1302.4956*, 2013.

[HSC18] Wenkai Hu, Sirish L Shah, and Tongwen Chen. Framework for a smart data analytics platform towards process monitoring and alarm management. *Computers & Chemical Engineering*, 114:225–244, 2018.

[LCH+19] Clement Lork, Vishal Choudhary, Naveed Ul Hassan, Wayes Tushar, Chau Yuen, Benny Kai Kiat Ng, Xinyu Wang, and Xiang Liu. An ontology-based framework for building energy management with iot. *Electronics*, 8(5):485, 2019.

[LHL+16] Thuc Duy Le, Tao Hoang, Jiuyong Li, Lin Liu, Huawen Liu, and Shu Hu. A fast pc algorithm for high dimensional causal discovery with multi-core pcs. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(5):1483–1495, 2016.

[Lin11] Morten Lind. An introduction to multilevel flow modeling. *Nuclear safety and simulation*, 2(1):22–32, 2011.

[Liz14] Joseph T Lizier. Jidt: An information-theoretic toolkit for studying the dynamics of complex systems. *Frontiers in Robotics and AI*, 1:11, 2014.

[LÖC07] Jan Eric Larsson, Bengt Öhman, and Antonio Calzada. Real-time root cause analysis for power grids. In *Security and Reliability of Electric Power Systems, CIGRE Regional Meeting, Tallinn, Estonia*, 2007.

[Maz09] Lawrence J Mazlack. Representing causality using fuzzy cognitive maps. In *NAFIPS 2009-2009 Annual Meeting of the North American Fuzzy Information Processing Society*, pages 1–6. IEEE, 2009.

[MDE+19] Ralf Mosshammer, Konrad Diwold, Alfred Einfalt, Julian Schwarz, and Benjamin Zehrfeldt. Bifrost: A smart city planning and simulation tool. In *International Conference on Intelligent Human Systems Integration*, pages 217–222. Springer, 2019.

[MDEG18] Ralf Mosshammer, Konrad Diwold, Alfred Einfalt, and Christoph Groiss. Reactive operation: a framework for event driven low voltage grid operation. In *International Conference on Intelligent Human Systems Integration*, pages 83–88. Springer, 2018.

[MGC+16] Nasrin Mostafazadeh, Alyson Grealish, Nathanael Chambers, James Allen, and Lucy Vanderwende. Caters: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events*, pages 51–61, 2016.

[MSTS14] Paramita Mirza, Rachele Sprugnoli, Sara Tonelli, and Manuela Speranza. Annotating causality in the tempeval-3 corpus. In *EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pages 10–19. Association for Computational Linguistics, 2014.

[NJZ+18] Emil Krabbe Nielsen, Stefan Jespersen, Xinxin Zhang, Ole Ravn, and Morten Lind. On-line fault diagnosis of produced water treatment with multilevel flow modeling. *Ifac-papersonline*, 51(8):225–232, 2018.

[OWBP16] Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, 2016.

[Pla18] André Platzer. *Logical foundations of cyber-physical systems*, volume 662. Springer, 2018.

[PSL14] Joern Ploennigs, Anika Schumann, and Freddy Lécué. Adapting semantic sensor networks for smart building diagnosis. In *International Semantic Web Conference*, pages 308–323. Springer, 2014.

[QDY+20] Juan Qiu, Qingfeng Du, Kanglin Yin, Shuang-Li Zhang, and Chongshu Qian. A causality mining and knowledge graph based method of root cause diagnosis for performance anomaly in cloud applications. *Applied Sciences*, 10(6):2166, 2020.

[RH04] James J Rooney and Lee N Vanden Heuvel. Root cause analysis for beginners. *Quality progress*, 37(7):45–56, 2004.

[RK02] Mario Ragwitz and Holger Kantz. Markov models from data by simple nonlinear time series predictors in delay embedding spaces. *Physical Review E*, 65(5):056201, 2002.

[Sch00] Thomas Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.

[SFSS09] Ansgar Scherp, Thomas Franz, Carsten Saathoff, and Steffen Staab. F–a model of events based on the foundational ontology dolce+ dns ultralight. In *Proceedings of the fifth international conference on Knowledge capture*, pages 137–144, 2009.

[SGSH00] Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000.

[SNDS90] Jerzy Splawa-Neyman, Dorota M Dabrowska, and TP Speed. On the application of probability theory to agricultural experiments. essay on principles. section 9. *Statistical Science*, pages 465–472, 1990.

[WRH+12] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[YDSC14] Fan Yang, Ping Duan, Sirish L Shah, and Tongwen Chen. *Capturing connectivity and causality in complex industrial processes*. Springer Science & Business Media, 2014.

[Zha12] Qin Zhang. Dynamic uncertain causality graph for knowledge representation and reasoning: Discrete dag cases. *Journal of Computer Science and Technology*, 27(1):1–23, 2012.