TU WIEN Informatics

On Automated Theorem Proving for Assertional and Refutational Natural Deduction Systems

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Michael Maurer
Registration Number 01634044

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Mag.rer.nat. Dr.techn. Hans Tompits

Vienna, 5th December, 2022

Michael Maurer	Hans Tompits

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

Erklärung zur Verfassung der Arbeit

Michael Maurer

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, 5. Dezember 2022

Michael Maurer

Acknowledgements

My gratitude goes to all of the many people who helped bring this thesis to fruition, most directly my supervisor Ao.Univ.Prof. Hans Tompits, whose suggestions and feedback always kept me on track and ensured I knew what to do. I would also especially like to thank my family and friends for remaining supportive and encouraging, and for giving me the opportunity to devote the necessary time to this work. Finally, as my years as a student draw to a close, I want to thank the general educational framework of the TU Wien, which allowed me to obtain both an interest in this fascinating topic and the knowledge to even begin tackling it.

Abstract

We investigate the use of automated theorem proving (ATP) techniques with regards to two natural deduction systems for classical propositional logic. One is the well-known assertional system following Gentzen and Jaśkowski, and the other is a refutational system due to Tamminga. Based on the approaches used by existing ATPs for natural deduction in the literature, we establish first a variant of the refutational system for which a weak normalisation theorem holds, and then variants of both systems that ensure loop-free derivations. We further implement and evaluate Hermes, a Prolog-based ATP that searches for assertional and refutational natural deduction derivations in parallel. Our observations show that this program is capable of producing remarkably human-like derivations for both valid and refutable formulas, but its performance, particularly in the refutational component, is not optimal due to the intensive backtracking required on certain failed derivation paths.

Kurzfassung

Diese Arbeit befasst sich mit den Einsatz von Techniken der automatisierten Beweisführung ("automated theorem proving", kurz ATP) im Bezug auf zwei Systeme des natürlichen Schließens für die klassische Aussagenlogik. Einerseits handelt es sich dabei um das weithin bekannte assertive Beweissystem nach Gentzen und Jaśkowski, andererseits um ein dazu komplementäres Verwerfungssystem nach Tamminga. Den Ansätzen folgend, die sich in der Literatur bei bereits existierenden ATPs für natürliches Schließen finden, erstellen wir zuerst eine Variante des Verwerfungssystems für die ein schwaches Normalisierungstheorem gilt, gefolgt von Varianten beider Systeme mit denen die Schleifenfreiheit der Beweise sichergestellt ist. Weiters implementieren und evaluieren wir Hermes, einen Prolog-ATP der parallel nach Assertions- und Verwerfungsbeweisen sucht. Aus unseren Beobachtungen geht hervor, dass sich mit einem solchen Programm durchaus Beweise generieren lassen, die den von Menschen erstellten ähneln, jedoch ist die Performanz insbesondere beim Generieren von Verwerfungsbeweisen nicht optimal, da gewisse gescheiterte Suchpfade intensives Backtracking erforderlich machen.

Contents

CHAPTER 1

Introduction

Automated theorem provers (ATPs) are programs which, given a logical statement, attempt to find a sequence of reasoning steps establishing the validity of that statement, with little to no help from their human users. The applications for such systems are broad, ranging from simply finding proofs of mathematical theorems to verifying the correctness of programs, or even generating programs to match a certain specification [10, 33, 14]. The basis for ATPs is formed by *proof calculi* which formalise the means of reasoning within a certain logic. In particular, the *resolution calculus* [32] has seen much success in the area of automated theorem proving, as its simplicity lends itself to efficient implementation [27]. Notable examples of resolution-based provers include Vampire [31] and E [35].

A downside of this development towards computing derivations in simple, compact calculi like resolution is an often exceedingly large number of intermediate steps in the output which a human who reasons about the same problem would not need to bother with, making them difficult to read and interpret [30]. With rising demand for systems of *explainable artificial intelligence* [34] that allow the user to understand the reasoning process behind every decision, this limitation is especially significant, as it hinders the application of ATPs in this area unless they are paired with some sort of translation interface. A potential solution lies in the use of calculi designed specifically for human-readability, for example a calculus in the style of *natural deduction* [18, 12]. Indeed, some theorem provers for first-order logic utilising natural deduction systems were already developed in the late 20th century [28, 2, 26, 23], and while such provers generally perform worse than their well-established resolution counterparts, they have demonstrated greater suitability for certain classes of problems [24].

The main contribution of this thesis is an ATP that produces derivations in a human-readable style of natural deduction, but at the same time, we address a second shortcoming of current state-of-the-art ATPs. Namely, they focus solely on proving *validity* or *unsatisfiability*, while their ability to determine *satisfiability* or *non-validity*, i.e., *refutability*,

is generally limited to utilising SAT solving algorithms or saturating the search space—approaches that may not always succeed and do not produce the kind of reasoning one would expect from an automated theorem prover. We will instead be making use of a much more direct way of proving refutability (and dually, statisfiability), namely a *refutation system*, or *complementary calculus*.

The latter kind of calculus is a formal proof system structured much like a traditional (assertional) calculus, but instead describes means of deducing the refutability of a statement. The idea of complementary calculi goes back to Jan Łukasiewicz, who first introduced them in connection with his work on Aristotelean syllogisms [20]. Refutation systems have subsequently been introduced for a variety of different logics, like intuitionistic logic [4], modal logics [13], and finite-valued logics [22, 1]. However, despite that the term "non-theorem prover" already appears in a work on the topic by Tiomkin in 1988 [40], the automation of complementary systems has so far only been explored by means of a Prolog implementation of such calculi for three-valued logics due to Oetsch and Tompits [22], as well as with a natural deduction proof-search procedure by Ferrari and Fiorentini [5] to obtain countermodels when the assertional proof fails.

The system we introduce in this thesis, Hermes, is an ATP for classical propositional logic which utilises both a traditional, assertional system of natural deduction and a complementary, refutational system of natural deduction to obtain a human-readable proof of either validity or refutability. The assertional calculus follows the original formulations of Gentzen and Jaśkowski, while the refutational calculus is due to Tamminga [39]. Our implementation builds on previous ATPs based on natural deduction by reusing strategies found in those systems, extending them also to the refutational case due to the similar structure of both calculi. A final evaluation of Hermes on selected propositional problems measures not only the general performance of these strategies, but also how similar the generated output truly is to a human-made derivation.

CHAPTER 2

Background

First of all, we must establish some fundamental definitions and notational conventions regarding the type of logic we work with and the basic natural deduction calculi for assertional and refutational reasoning.

2.1 Classical Propositional Logic

2.1.1 Syntax

Definition 1. The *alphabet* of CPL consists of a set $\mathcal{P} = \{p_0, p_1, p_2, p_3, \dots\}$ of *propositional constants*, the *logical connectives* $\wedge$ ("conjunction"), $\vee$ ("disjunction"), $\to$ ("implication"), $\neg$ ("negation"), $\top$ ("verum"), and $\bot$ ("falsum"), and the *parentheses* "(" and ")". □

Definition 2. The set $\mathcal{A}$ of *atomic formulas* is given by

$$\mathcal{A} = \mathcal{P} \cup \{\top, \bot\}.$$

□

Definition 3. The set $\mathcal{F}$ of *propositional formulas* is the smallest set subject to the following conditions:

1. $\mathcal{A} \subseteq \mathcal{F}$;
2. if $F_1, F_2 \in \mathcal{F}$ and $\circ \in \{\wedge, \vee, \to\}$, then $(F_1 \circ F_2) \in \mathcal{F}$; and
3. if $F \in \mathcal{F}$, then $\neg F \in \mathcal{F}$.

□

Definition 4. The *degree* of a propositional formula $F \in \mathcal{F}$, symbolically $\deg(F)$, is given by

$$\deg(F) = \begin{cases} 0, & \text{if } F \in \mathcal{A}, \\ \deg(F') + 1, & \text{if } F = \neg F', \\ \max(\deg(F_1), \deg(F_2)) + 1, & \text{if } F = F_1 \circ F_2, \circ \in \{\wedge, \vee, \rightarrow\}. \end{cases}$$

□

2.1.2 Semantics

Definition 5. An *intepretation* I is a function whose domain is some subset $\mathcal{P}_I$ of the propositional constants and whose range is $\{0, 1\}$. □

Definition 6. Let I be an interpretation. Then, the *valuation based on* I, V_I, is a function that has as its domain the set of all formulas containing only propositional constants from $\mathcal{P}_I$ and as its range $\{0, 1\}$, and such the following conditions hold:

- $V_I(F) = I(F)$, if $F \in \mathcal{P}_I$;
- $V_I(\bot) = 0$;
- $V_I(\top) = 1$;
- $V_I(\neg F) = 1 - V_I(F)$;
- $V_I(F_1 \wedge F_2) = \min(V_I(F_1), V_I(F_2))$;
- $V_I(F_1 \vee F_2) = \max(V_I(F_1), V_I(F_2))$;
- $V_I(F_1 \rightarrow F_2) = 0$ if $V_I(F_1) = 1$ and $V_I(F_2) = 0$, and $V_I(F_1 \rightarrow F_2) = 1$ otherwise.

□

If $V_I(F) = 1$, then the interpretation I makes the formula F *true* (or F is *true in* I) and is a *model* of F. Conversely, if $V_I(F) = 0$, we say that I makes F *false* (or F is *false in* I) and is a *countermodel* of F. A formula F is *valid*, symbolically $\models F$, if F is true in every interpretation.

An interpretation is a model of a set $\Gamma \subseteq \mathcal{F}$ of formulas if it is a model of each element of Γ, otherwise it is a countermodel.

Definition 7. A formula F is *entailed* by set $\Gamma \subseteq \mathcal{F}$ of formulas, symbolically $\Gamma \models F$, if every model of Γ is also a model of F. □

More generally, for a set $\Delta \subseteq \mathcal{F}$ of formulas, Δ is entailed by Γ, symbolically $\Gamma \models \Delta$, if every model of Γ makes at least one formula of Δ true.

The inverse concept to entailment is *refutation*, which is central in the context of complementary calculi.

Definition 8. A formula F is *refuted* by a set $\Gamma \subseteq \mathcal{F}$ of formulas, symbolically $\Gamma \nmodels F$, if there is an interpretation which is a model of every formula in Γ but a countermodel of F. □

The more general form $\Gamma \nmodels \Delta$ then means that there is an interpretation which is a model of every formula in Γ and a countermodel of every formula in Δ. Clearly, $\Gamma \models \Delta$ holds exactly when $\Gamma \nmodels \Delta$ does not.

2.2 Two Natural Deduction Systems

2.2.1 Preliminaries

The purpose of natural deduction calculi, first introduced independently by Gerhard Gentzen [12] and Stanislaw Jaśkowski [18] in 1934, is to formally recreate those patterns of reasoning commonly employed by human mathematicians. Their most characteristic feature is the ability to make an arbitrary *assumption*, perform a *subdeduction* in which that assumption is treated like an already derived formula, and finally *discharge* the assumption to gain new information that holds independently of it. This simple paradigm allows for the straightforward recreation of techniques like *proof by contradiction* or *reasoning by cases*.

The basic structure remains the same in both assertional and refutational natural deduction systems. New formulas are introduced into a derivation either through the use of *inference rules* or by making *assumptions*. An inference rule consists of zero or more *premisses* and a single *conclusion*, the former being formulas that must already have been derived to apply the rule and the latter being the formula obtained by doing so. In some cases, *side conditions* may further restrict when it is possible to apply a rule. In natural deduction, inference rules are commonly categorised as either *introduction rules* (whose conclusion introduces a connective not present in the premisses) or *elimination rules* (whose conclusion eliminates a connective present in the premisses), although certain rules may also fall in neither category.

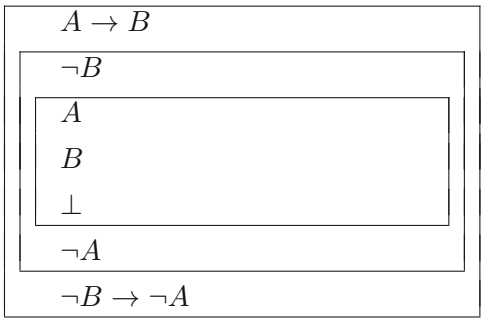
Assumptions are arbitrary formulas that can be *introduced* at will throughout a derivation, and the most recently made assumption can be *discharged* at any point as well. The portion of the derivation between introduction and dischargement of an assumption forms a *subdeduction* in which the assumption is treated just like any other derived formula, and the inside of the subdeduction is considered to be in the *scope* of that assumption. Since it is always allowed to introduce an assumption, subdeductions can, of course, be nested.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \to B}\,u \quad \dfrac{}{A}\,w}{B}\,\to E \quad \dfrac{}{\neg B}\,v}{\bot}\,\neg E}{\neg A}\,\neg I^{w}}{\neg B \to \neg A}\,\to I^{v}}{(A \to B) \to (\neg B \to \neg A)}\,\to I^{u}$$

Gentzen (1935)

1. $A \to B$
1.1. $\neg B$
1.1.1. A
1.1.1. B
1.1.1. $\bot$
1.1. $\neg A$
1. $\neg B \to \neg A$
$(A \to B) \to (\neg B \to \neg A)$

Jaśkowski (1934)



$(A \to B) \to (\neg B \to \neg A)$

Jaśkowski (1929) [17]

Figure 2.1: Sample derivations in Gentzen- and Jaśkowski-style deduction.

While a formula F derived within the scope of an assumption A generally becomes unavailable for use in the premisses of inference rules after A has been discharged, it is possible for a rule to require F derived specifically in the scope of A, effectively making the premiss a subdeduction beginning with A and ending on F (we thus also refer to such premisses as *subdeduction premisses*). Through the use of rules with subdeduction premisses, making assumptions allows us to derive new information that does not depend on those assumptions.

To keep track of assumptions and their scopes, several different types of notations have been used in the history of (assertional) natural deduction calculi. Figure 2.1 compares those used in the foundational works of Gentzen and Jaśkowski on the example of a derivation of $(A \to B) \to (\neg B \to \neg A)$.

1.	$A \to B$	assumption
2.	$\neg B$	assumption
3.	A	assumption
4.	B	$\to E$: 1, 3
5.	$\bot$	$\neg E$: 2, 4
6.	$\neg A$	$\neg I$: 2, 3–5
7.	$\neg B \to \neg A$	$\to I$: 2–6
8.	$(A \to B) \to (\neg B \to \neg A)$	$\to I$: 1–7

Figure 2.2: Sample derivation in our notation.

Gentzen [12] depicted derivations as a tree-like structure having the final conclusion at the root and assumptions at the leaves, with each assumption being labeled with a letter on introduction that is also noted next to the rule that discharges it (by using the subdeduction it started as its premiss). Jaśkowski [18] simply wrote the formulas occuring in a derivation line by line and indicated assumption scopes, as well as the nesting of subdeductions, with a number to the left of each formula. Also in Jaśkowski's work, we can find a variant of this notation he used previously in 1929, using simple boxes drawn around groups of lines in the derivation to show the subdeductions. Similar "boxed" notations later saw significant use by other authors [8, 9], and we too will be using this type of notation for the assertional and refutational systems described in the present chapter.

Our notation slightly extends Jaśkowski's from 1929 through the addition of a column to the left that simply tracks the line number, and a column to the right that shows by what means each formula was derived and at which preceding lines the premisses can be found. If a premiss is a subdeduction, its origin is written as i–j, where i and j are the first and the final line of the subdeduction, respectively. As a result, the derivation of $(A \to B) \to (\neg B \to \neg A)$ now takes the form seen in Figure 2.2.

Inference rules will be specified in a similar way, for example the assertional introduction rule for $\to$ is given as follows:

$$\dfrac{\begin{array}{|cc|}\hline \vdots & \\ m & A \\ & \vdots \\ n-1 & B \\ \hline\end{array}}{\begin{array}{cc} n & A \to B \end{array}} \to I$$

Since the premiss is a subdeduction, it is shown in a box ranging from assumption A at

line m to the formula B at line $n-1$. Inbetween, and also before the premiss, the symbol

$$\vdots$$

represents an arbitrary derivation of zero or more lines, subject to the condition that no previously introduced assumptions are discharged within those lines. It is therefore guaranteed that B is still in the scope of A, and in this case there can also be no assumptions made after A still active, since we must discharge A after line $n-1$ to apply the rule. On the other hand, both subdeduction and conclusion may be nested within any number of other assumptions, introduced (but not discharged) within the placeholder before line m.

2.2.2 The Assertional Calculus N

In what follows, we will describe the inference rules that together form the assertional natural deduction calculus denoted by **N**.

The simplest and most trivial rule is *Triv*, which lets us repeat a formula that has been derived at a preceding line:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & F \\ & \vdots \end{array}}{\begin{array}{ll} n & F \end{array}}\ \mathit{Triv}$$

This rule is useful in order to increase the flexibility of other rules and reflects the human process of recalling an assumption or an intermediate result, thus aiding a more natural reasoning style.

Another common technique used by humans is a proof by contradiction, where an assumption is made only to derive a contradiction within its scope and thereby showing that the assumption must be false. In **N**, this type of reasoning is handled by the rule *CRF* ("classical reductio ad falsum") and by the rule $\neg I$, the introduction rule for the $\neg$ connective:

$$\dfrac{\begin{array}{ll} & \vdots \\ \boxed{\begin{array}{ll} m & \neg F \\ & \vdots \\ n-1 & \bot \end{array}} & \end{array}}{\begin{array}{ll} n & F \end{array}}\ \mathit{CRF} \qquad \dfrac{\begin{array}{ll} & \vdots \\ \boxed{\begin{array}{ll} m & F \\ & \vdots \\ n-1 & \bot \end{array}} & \end{array}}{\begin{array}{ll} n & \neg F \end{array}}\ \neg I$$

Contradictions are indicated by $\bot$, as it is false in every interpretation and should not be derivable unless our assumptions are flawed.

Introduction rules exist for all other connectives as well. In the case of $\wedge$, we need both of the subformulas occurring to the left and to the right of $\wedge$ as premisses:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & A \\ & \vdots \\ n-1 & B \end{array}}{\begin{array}{ll} n & A \wedge B \end{array}} \wedge I_1 \qquad \dfrac{\begin{array}{ll} & \vdots \\ m & B \\ & \vdots \\ n-1 & A \end{array}}{\begin{array}{ll} n & A \wedge B \end{array}} \wedge I_2$$

For $\vee$, on the other hand, either subformula is sufficient by itself:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & A \\ & \vdots \end{array}}{\begin{array}{ll} n & A \vee B \end{array}} \vee I_1 \qquad \dfrac{\begin{array}{ll} & \vdots \\ m & B \\ & \vdots \end{array}}{\begin{array}{ll} n & A \vee B \end{array}} \vee I_2$$

The $\rightarrow$ introduction rule is especially elegant and uses assumptions in perhaps the most straightforward and intuitive manner:

$$\dfrac{\boxed{\begin{array}{ll} & \vdots \\ m & A \\ & \vdots \\ n-1 & B \end{array}}}{\begin{array}{ll} n & A \rightarrow B \end{array}} \rightarrow I$$

Even $\top$ possesses a dedicated introduction rule, but it requires no premisses since a formula that is always true should be derivable no matter the context:

$$\dfrac{\begin{array}{ll} & \vdots \end{array}}{\begin{array}{ll} n & \top \end{array}} \top I$$

The other major category are elimination rules, which again exists for each connective. The pair of $\wedge$ elimination rules allows us to simply extract the subformula from either side of $\wedge$, since they must both be true:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & A \wedge B \\ & \vdots \end{array}}{\begin{array}{ll} n & A \end{array}} \wedge E_1 \qquad \dfrac{\begin{array}{ll} & \vdots \\ m & A \wedge B \\ & \vdots \end{array}}{\begin{array}{ll} n & B \end{array}} \wedge E_2$$

While their introduction rules were quite similar, $\vee$ takes quite a different path from $\wedge$ when it comes to elimination rules, implementing another common human proof approach

by splitting the reasoning into two cases (i.e., subdeductions) that must both reach the same conclusion:

$$\dfrac{\begin{array}{ll} & \vdots \\ k & A \vee B \\ \boxed{\begin{array}{ll} k+1 & A \\ & \vdots \\ m & F \end{array}} & \\ \boxed{\begin{array}{ll} m+1 & B \\ & \vdots \\ n-1 & F \end{array}} & \\ n & F \end{array}}{\vee E}$$

The premiss at line k is referred to as the *major premiss*, and $A \vee B$ as the *major formula*, since it contains $\vee$, the characteristic connective of the rule. Correspondingly, the two subdeductions are *minor premisses* and the two copies of F are *minor formulas*.

The connective $\rightarrow$ continues to receive very intuitive rules, in this case following the basic reasoning technique of *modus ponens* to extract B from a major formula $A \rightarrow B$ via a minor formula A:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & A \rightarrow B \\ & \vdots \\ n-1 & A \\ n & B \end{array}}{\rightarrow E}$$

The contradictions we require in order to apply *CRF* and $\neg I$ are most easily obtained using the elimination rule for $\neg$, which takes a pair of contradictory formulas as its input and concludes $\bot$ from them:

$$\dfrac{\begin{array}{ll} & \vdots \\ k & \neg F \\ & \vdots \\ m & F \\ & \vdots \\ n & \bot \end{array}}{\neg E}$$

Another way to use $\bot$, should it ever be encountered, is by using its own elimination rule to derive an arbitrary formula, following the principle of *ex falso quodlibet*:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & \bot \\ & \vdots \\ n & F \end{array}}{\bot E}$$

Definition 9. A formula F is *provable in* **N** *under assumptions* Γ, symbolically $\Gamma \vdash F$, if there exists a derivation using the rules in **N** whose final line contains F and is in the scope of exactly all assumptions in Γ. □

Theorem 1. *The calculus* **N** *is* sound*, i.e., if* $\Gamma \vdash F$*, then* $\Gamma \models F$*.*

Theorem 2. *The calculus* **N** *is* complete*, i.e., if* $\Gamma \models F$*, then* $\Gamma \vdash F$*.*

Soundness and completeness proofs for assertional natural deduction systems can be readily found in the literature, like, e.g., in the well-known textbook by Huth and Ryan [16] (in fact, a proof for specifically the system **N** discussed here is detailed in the bachelor's thesis of the current author [21]).

2.2.3 The Refutational Calculus $\overline{\mathbf{N}}$

The refutational calculus $\overline{\mathbf{N}}$ we introduce now is an adaption of a system discussed by Allard M. Tamminga [39]. Its inference rules are quite similar to those found in **N**, but before introducing them, we require some additional concepts regarding the assumption scopes formulas can be in.

Definition 10. Let m and n ($m \leq n$) be line numbers in a derivation used for $\overline{\mathbf{N}}$ and F_m the formula written on line m. We say that

(i) F_m is *operative at line* n if line m is within the scope of at least one assumption, and line n is in the scope of all assumptions in whose scope line m is (i.e., lines m and n share a box);

(ii) F_m is *accessible at line* n if neither line m nor line n are in the scope of any assumptions; and

(iii) F_m is *attainable at line* n if it is operative or accessible at line n.

□

A universal side condition for all rules of $\overline{\mathbf{N}}$ is that any premisses used must be attainable at the line where the conclusion is written, with premisses of a subdeduction being treated as belonging to the scope that contains them as a whole (since otherwise the additional assumption would prevent their use entirely). Specifically, this means that, in contrast to **N**, conclusions made without any assumptions cannot be used for inferences made after introducing an assumption.

Being aware of this general restriction, we can now look at the rules themselves, beginning with the *Triv* rule of $\overline{\mathbf{N}}$:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & F \\ & \vdots \end{array}}{\begin{array}{ll} n & F \end{array}}\ \mathit{Triv}$$

Perhaps unsurprisingly, it does not differ at all from its assertional counterpart, as the act of reusing already known information is independent of what we are ultimately trying to prove.

On the other hand, a significant difference between **N** and $\overline{\mathbf{N}}$ lies in the following two rules exclusive to $\overline{\mathbf{N}}$, together known as the *atomic rules*:

$$\frac{\begin{array}{cc} & \vdots \\ n & p \end{array}}{} At \qquad \frac{\begin{array}{cc} & \vdots \\ n & \neg p \end{array}}{} At$$

Both rules are subject to the side condition that p does not occur in any formula operative at line n.

These rules allow us to freely introduce into the derivation any propositional constant p or its negation $\neg p$, under the stipulation that p has not yet been used anywhere in the operative formulas. This captures the simple fact that a lone propositional constant with no other formula imposing restrictions on it can be assigned either truth value by an interpretation, meaning both it and its negation always have countermodels and thus are refutable. Interestingly, the side condition does not apply to accessible formulas, making it possible to derive both p and $\neg p$ side-by-side outside the scope of all assumptions, but side conditions on other rules and the fact that these formulas cannot be used as premisses once an assumption has been made averts contradictions from arising.

More familiar again are the rules which allow using proofs by contradiction in refutational reasoning as well:

$$\frac{\begin{array}{|cc} & \vdots \\ m & \neg F \\ & \vdots \\ n-1 & \top \end{array}}{\begin{array}{cc} n & F \end{array}} CRV \qquad \frac{\begin{array}{|cc} & \vdots \\ m & F \\ & \vdots \\ n-1 & \top \end{array}}{\begin{array}{cc} n & \neg F \end{array}} \neg I$$

Where **N** indicated a contradiction with $\bot$, the formula that can never be true, $\overline{\mathbf{N}}$ uses $\top$, the formula that can never be false (and thus concluding its refutability points to a problem in the assumptions). Accordingly, we now have a rule named *CRV*, for *classical reductio ad verum*, paired with the $\neg$ introduction rule.

Continuing with the introduction rules, $\wedge$ simply requires deriving the refutability of either its immediate subformula, much like how $\vee$ works in the assertional case:

$$\frac{\begin{array}{cc} & \vdots \\ m & A \\ & \vdots \end{array}}{\begin{array}{cc} n & A \wedge B \end{array}} \wedge I_1 \qquad \frac{\begin{array}{cc} & \vdots \\ m & B \\ & \vdots \end{array}}{\begin{array}{cc} n & A \wedge B \end{array}} \wedge I_2$$

Correspondingly, the introduction rules for $\vee$ do not exactly mirror the assertional ones for $\wedge$ but instead feature a subdeduction for the second premiss:

$$\dfrac{\begin{array}{|ll}\multicolumn{2}{c}{\vdots}\\ m & A\\ \hline m+1 & A\\ & \vdots\\ n-1 & B\\ \hline\end{array}}{n \quad A \vee B}\ \vee I_1 \qquad \dfrac{\begin{array}{|ll}\multicolumn{2}{c}{\vdots}\\ m & B\\ \hline m+1 & B\\ & \vdots\\ n-1 & A\\ \hline\end{array}}{n \quad A \vee B}\ \vee I_2$$

Not only must both A and B be derived, but one of the two must also be the conclusion of a subdeduction having the other as its assumption. What may seem like a pointless detour at first glance is actually necessary so that applications of $\vee$ introduction rules outside the scope of all assumptions cannot be used to wrongfully establish the refutability of formulas such as $p \vee \neg p$.

For the $\rightarrow$ introduction rules, we make use of the equivalence between $A \rightarrow B$ and $\neg A \vee B$ to reuse the structure from above:

$$\dfrac{\begin{array}{|ll}\multicolumn{2}{c}{\vdots}\\ m & \neg A\\ \hline m+1 & \neg A\\ & \vdots\\ n-1 & B\\ \hline\end{array}}{n \quad A \rightarrow B}\ \rightarrow I_1 \qquad \dfrac{\begin{array}{|ll}\multicolumn{2}{c}{\vdots}\\ m & B\\ \hline m+1 & B\\ & \vdots\\ n-1 & \neg A\\ \hline\end{array}}{n \quad A \rightarrow B}\ \rightarrow I_2$$

Finally, $\bot$ takes the place of $\top$ as a formula that can unconditionally be introduced at any point, being always false by definition:

$$\dfrac{\begin{array}{l}\vdots\\ \hline\end{array}}{n \quad \bot}\ \bot I$$

On the elimination rule side, we can observe another reversal of $\wedge$ and $\vee$, with the former now having the rule that implements case distinctions:

$$\dfrac{\begin{array}{ll}\multicolumn{2}{c}{\vdots}\\ k & A \wedge B\\ \hline\multicolumn{2}{|l}{k+1 \quad A}\\ \multicolumn{2}{|l}{\vdots}\\ \multicolumn{2}{|l}{m \quad F}\\ \hline\multicolumn{2}{|l}{m+1 \quad B}\\ \multicolumn{2}{|l}{\vdots}\\ \multicolumn{2}{|l}{n-1 \quad F}\\ \hline\end{array}}{n \quad F}\ \wedge E$$

On the other hand, eliminating a formula with $\vee$ gives us direct access to its subformulas, and by the interpretation of $\rightarrow$ as a special case of $\vee$, we can use the same format of elimination rules for it as well:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & A \vee B \\ & \vdots \\ n & A \end{array}}{} \vee E_1 \qquad \dfrac{\begin{array}{ll} & \vdots \\ m & A \vee B \\ & \vdots \\ n & B \end{array}}{} \vee E_2$$

$$\dfrac{\begin{array}{ll} & \vdots \\ m & A \rightarrow B \\ & \vdots \\ n & \neg A \end{array}}{} \rightarrow E_1 \qquad \dfrac{\begin{array}{ll} & \vdots \\ m & A \rightarrow B \\ & \vdots \\ n & B \end{array}}{} \rightarrow E_2$$

Obtaining $\top$ is still mainly done through the elimination of contradictions, although there is an added restriction in the form of the side condition that the formulas $\neg F$ and F in the rule below are operative at line n:

$$\dfrac{\begin{array}{ll} & \vdots \\ k & \neg F \\ & \vdots \\ m & F \\ & \vdots \\ n & \top \end{array}}{} \neg E$$

This specifically means that $\neg E$ can never be used outside the scope of all assumptions, where formulas would only be accessible, and so we avoid the derivation of incorrect results from contradictions introduced using solely atomic rules with no assumptions.

Lastly, the $\top$ elimination rule lets us immediately declare any formula refutable in the presence of a contradiction:

$$\dfrac{\begin{array}{ll} & \vdots \\ m & \top \\ & \vdots \\ n & F \end{array}}{} \top E$$

Definition 11. A formula F is *refutable in* $\overline{\mathbf{N}}$ *under assumptions* Γ, symbolically $\Gamma \dashv F$, if there exists a derivation using the rules in $\overline{\mathbf{N}}$ whose final line contains F and is in the scope of exactly all assumptions in Γ. □

The adequacy of derivations in $\overline{\mathbf{N}}$ is not quite as direct a match between $\Gamma \dashv F$ and $\Gamma \nvDash F$. Indeed, $\dashv F$ holds exactly when $\nvDash F$ does, but a proof of this must also take operative formulas other than assumptions into account.

Definition 12. Let n refer to a line in a derivation in $\overline{\mathbf{N}}$. Then, $\mathfrak{R}_n$ denotes the set of formulas from lines $m < n$ in the derivation that are operative at line n, plus the formula at line n itself if it is an assumption. □

Theorem 3. *The calculus* $\overline{\mathbf{N}}$ *is* sound*, i.e., if* $\Gamma_n \dashv F_n$ *and* $\models \mathfrak{R}_n$*, then* $\models \mathfrak{R}_n \cup \{F_n\}$.

In particular, since $\Gamma_n = \emptyset$ means that no formulas are operative at line n and therefore $\mathfrak{R}_n = \emptyset$, it follows that, if $\dashv F_n$, then $\models F_n$.

Theorem 4. *The calculus* $\overline{\mathbf{N}}$ *is* complete*, i.e., if* $\models F_n$*, then* $\dashv F_n$.

Proofs of soundness and completeness for this complementary calculus can be found both in the original work by Tamminga [39] as well as in more detail in the author's aforementioned thesis [21].

CHAPTER 3

Automated Theorem Proving with Natural Deduction

Several automated theorem provers utilising forms of natural deduction have been implemented and described in the literature in the past. In order to discuss the design of the ATP introduced in this thesis, Hermes, we now investigate some of these systems, with particular focus on how they keep track of assumptions and which strategies guide their application of natural deduction rules. Given this focus, the systems of interest are those that use both assumptions in the style of natural deduction and rules similar to the ones found in the natural deduction calculi originating from Gentzen and Jaśkowski, including:

- Pollock's Oscar (1990) [28];
- Pelletier's Thinker (1992) [26];
- Dafa's ANDP (1992) [2];
- Sieg's and Byrnes's *ic-calculi* (1998) [36]; and
- Ferrari's and Fiorentini's proof search procedure for CPL (2015) [5] and IPL (2019) [6].

Not all of the above systems have actually been implemented as working ATPs, and we have omitted at least one major natural deduction ATP, Muscadet [23]. Both of these facts are because we are ultimately searching for descriptions of strategies that can be used in automated proof search with natural deduction, which are provided in detail by the selected works, and that is not available for Muscadet in any documentation we were able to locate.

3.1 Existing ATPs

3.1.1 Oscar

John Pollock's Oscar [28] is an automated theorem prover that implements human-like reasoning patterns. A central concept of this system is *interest-driven reasoning*, which means that inference rules are not used whenever they are applicable, but only if doing so goes towards the current "interest" or goal of the reasoning. The rules used by the system are standard introduction and elimination rules of natural deduction, but they are used in different ways: introduction rules appear in backwards reasoning form and are applied to derive the premisses from the conclusions, while elimination rules appear in (random) forward reasoning form and are used to derive conclusions from premisses.

The system is capable of both linear and non-linear reasoning, the latter of which most closely resembles natural deduction with the use of assumptions and will be the focus of our investigation.

For purely linear reasoning without assumptions, the proof procedure in Oscar utilises multiple data structures:

- Input: initially given formulas that are adopted as beliefs from the start;
- Ultimate: formulas to be ultimately proven;
- Adoption-Queue: holds formulas to be newly adopted as beliefs, where an Adoption-Queue element of the form $\langle F, \Pi \rangle$ denotes that formula F should be adopted on the basis of the formula set Π (i.e., there is a rule that allows deriving F from Π);
- Adoptions: formulas that have been adopted as beliefs;
- Interest-Queue: holds new goals in which to adopt interest, where an Interest-Queue element of the form $\langle F, \Pi_0, \Pi, G \rangle$ denotes that F is of interest in order to derive G from Π, with Π_0 denoting the set of unadopted formulas in $\Pi \setminus \{F\}$;
- Interests: formulas that have been adopted as interests;
- Forset: keeps track of why the program is interested in formulas that have been adopted as interests, using Interest-Queueelements; and
- Basis: holds a record of the reasoning that is later used to produce a deductive proof.

After being initialized with Input and Ultimate, the steps for linear reasoning are simply

1. For each $F \in$ Ultimate, add $\langle F, \emptyset, \emptyset, \emptyset \rangle$ to Interest-Queue.

2. For each $F \in$ Input, add F to Adoptions.
3. Repeat until Interest-Queue and Adoption-Queue are both empty:
   a) If Interest-Queue is not empty, process it.
   b) If Interest-Queue is empty and Adoption-Queue is not, process Adoption-Queue.

When processing Interest-Queue, its first element $\langle F, \Pi_0, \Pi, G\rangle$ is considered. If neither F nor $\overline{F}$ (for $F = \neg F'$, $\overline{F}$ is F', otherwise $\neg F$) have been adopted as beliefs, this element is moved to Forset and we adopt interest in F. If $F \in$ Adoptions, we instead adopt interest in the first unadopted $H \in \Pi_0$, or, if there are none, insert $\langle G, \Pi\rangle$ into Adoption-Queue.

When adopting interest in a formula F, we also search the backwards reasoning rules for one that can derive F from some set of formulas Π. If such a rule and Π are found, we either immediately insert $\langle F, \Pi\rangle$ into Adoption-Queue if $\Pi \subseteq$ Adoptions (as we can immediately apply the rule in that case), or otherwise insert $\langle H, \Pi_0 \setminus \{H\}, \Pi, F\rangle$ into Interest-Queue, where Π_0 are the unadopted members of Π and H the first element of Π_0.

Adoption-Queue is processed by moving the first member $\langle F, \Pi\rangle$ to Basis and then applying the following steps in order:

1. Insert F into Adoptions.
2. Delete F from Ultimate.
3. If Ultimate empty, terminate.
4. Discharge interest in F. This involves taking every $\langle F, \Theta_0, \Theta, G\rangle$ in Forset and removing it. We also check the unadopted members of Θ_0, denoted Θ^*, and either insert $\langle H, \Theta^* \setminus \{H\}, \Theta, G\rangle$ into Interest-Queue for the first $H \in \Theta^*$ or $\langle G, \Theta\rangle$ into Adoption-Queue if there is no such H.
5. Check if adopting F completes the premisses Θ for any forwards reasoning rule with conclusion G, and if so, insert $\langle G, \Theta\rangle$ into Adoption-Queue.
6. Cancel interest in F (see below for details on cancelling interest).
7. If $\overline{F} \notin$ Ultimate, cancel interest in $\overline{F}$.

Cancelling interest in a formula F is a slightly sophisticated procedure that recursively cleans up the interest in F as well as any formulas we were only interested in as a way to get to F:

1. Delete F from Interests and Ultimate.

2. Delete any elements of Interest-Queue with F as their first or last member.

3. Delete any elements of Forset with F as their last member. If the first member G of such a deleted element now occurs neither in Ultimate nor as the first member of any remaining Forset elements, cancel interest in G.

A notable strategic choice seen in both the application of backwards reasoning rules and the interest discharging procedure is that additional formulas needed to fulfill an interest as recorded in Forset are inserted into Interest-Queue one at a time, rather than all at once. This means that when one of these required formulas cannot be proven, the program will not waste time adopting interest in the others and simply terminate with undischarged interests once the queues are fully processed.

So far, we have described only linear reasoning, but the extension to non-linear reasoning is quite easy. The simplest form of this kind of reasoning supported by Oscar is conditionalisation, which corresponds to the $\to I$ rule of our natural deduction system **N**. This requires reasoning within the scope of certain (possibly nested) assumptions, which is achieved by simply giving each assumption scope A its own copy of the various data structures—for example, Adoptions A holds the formulas that have been adopted within A (and surrounding assumptions). A new data structure is Suppose A, which contains the assumed formulas of the scopes up to and including A—different from plain natural deduction, Oscar also allows for multiple formulas to be assumed in one and the same scope (which is equivalent to assuming the conjunction of all those formulas in **N**). The outermost scope is also technically considered an assumption, called Beliefs, and Suppose Beliefs is empty.

Assumptions are created with both formulas to assume and formulas we aim to derive under those assumptions, and Ultimate, Interest-Queue, and Adoption-Queue are set up accordingly. However, if the formulas to assume are Γ and there already is an assumption scope A so that (Suppose A) $= \Gamma$, the data structures of that existing scope are extended instead of introducing the assumptions again.

Using assumptions also changes the form of elements in Interest-Queue and Forset to $\langle F, \alpha, \Pi_0, \Pi, G, A\rangle$. Here, α is used to mark which entries come from linear reasoning rules ($\alpha = \mathit{con}$) and which from conditionalisation ($\alpha = \mathit{condit}$) and A is the assumption scope in which we want to derive G. Π now consists of elements $\langle H, A^*\rangle$, with A^* being the assumption scope where we need H. Π_0 are those $\langle H, A^*\rangle \in \Pi$ so that $H \notin$ Adoptions A^*, except for $\langle G, A\rangle$.

Conditionalisation is applied to formulas of the form $F_1 \to F_2$ and begins by creating an assumption A^* starting with F_1 and aiming for F_2. Then, $\langle F_2, \mathit{condit}, \emptyset, \{F_2, A^*\}, (F_1 \to F_2), A\rangle$ is added to Interest-Queue A^* and processing continues in A^*.

With the addition of assumptions, discharging interest is extended in order to allow discharging assumptions once they have served their purpose. Recall that discharging interest in a formula F in the linear reasoning case meant also removing the elements

of FORSET with F as their first member and either adopting interest in the first of the remaining formulas needed to deduce the last member G or adopting G if no such formulas are left. This procedure remains the same for non-linear reasoning, but afterwards processing continues in the scope of the assumption matching whatever formula we adopted or adopted interest in. In particular, this means that as soon as we adopt the formula for which the assumption was originally made, the scope of that assumption is exited.

Another type of non-linear reasoning the program is capable of is reductio ad absurdum, or proof by contradiction. We create an assumption $\overline{F}$ and try to derive F within its scope, which also allows concluding it outside. A more complete technique is finding a contradiction not between the assumption $\overline{F}$ and F, but between an arbitrary formula and its negation. The program also attempts to find these contradictions by automatically adopting interest in the negation of anything it adopts while in the scope of an assumption made for reductio ad absurdum. To avoid overly deep nesting of such assumptions, reductio ad absurdum is only used as a last resort.

Finally, reasoning by cases is applied when a formula $F_1 \vee F_2$ is adopted. First an assumption is created with F_1, then once a new formula is adopted there we create another assumption with F_2 (but importantly not as a nested assumption) and try to derive the same formula there. This gives us a situation where we have already adopted $F_1 \vee F_2$ and the newly derived formula follows from F_1 and F_2 independently, so we can adopt that formula in the outer scope in the same fashion as the $\vee E$ rule of **N**.

The overall control structure is to linearly process the active assumption (denoted HOME, where initially HOME = BELIEFS) until one of the following conditions triggers a move to another assumption:

- Both INTEREST-QUEUE HOME and ADOPTION-QUEUE HOME are empty, so there is nothing left to do and we exit to the containing scope.
- Some $F \in$ ULTIMATE HOME is adopted, i.e., the assumption reaches the goal it was made for and we can continue to derive something new outside its scope.
- A new assumption is created due to a non-linear reasoning rule, so we continue processing in the scope of that assumption.

Oscar is written in Common LISP.

3.1.2 Thinker

Thinker [26] was developed by Francis Pelletier in the 1980s as a system to generate and present derivations in Fitch-style natural deduction [8], specifically in a system for first-order logic with identity due to Kalish and Montague [19] (as our aim is to build a prover for propositional logic only, we will focus on the portions relevant to that fragment). This system provides introduction and elimination rules for each connective,

plus rules to handle quantifiers and identity, and as a natural deduction system, it allows for the introduction and dischargement of assumptions in order to form subdeductions.

The way such subdeductions are presented is special: they always begin with a line of the form "show F", where F is the formula the subdeduction is meant to derive. Only when this has been successfully done and the subdeduction has been completed is F actually treated as a formula that has been deduced (an *antecedent formula*). The first line after the "show F" is usually an assumption, but only the following specific assumptions are allowed depending on F:

- If F is of the form $F_1 \to F_2$, the assumption can be F_1.
- If F is of the form $\neg F_1$, the assumption can be F_1.
- The assumption can always be $\neg F$.

To make F available as an antecedent formula that other rules can use for further deduction, we need to complete the subdeduction and *cancel* the "show" line. There are several conditions under which this is possible, two of which are relevant to propositional logic:

- for any F, "show F" can be cancelled if the subdeduction directly (and not within a nested subdeduction) contains the formula F, or contains both F' and $\neg F'$ for some formula F' (i.e., a contradiction);
- "show $F_1 \to F_2$" can be cancelled if the subdeduction directly contains F_2.

Completing a subdeduction makes it so that the lines following "show F" are *boxed* and no longer antecedent, while the "show" line itself is cancelled and F does become antecedent.

The system is implemented using three main data structures:

(i) Proofmatrix holds the produced derivation, including information such as justifications for each step or indications of boxing and cancelling.

(ii) Goalstack is simply a stack of (sub)goals that are to be shown, with the topmost element always being the one Thinker is currently working on.

(iii) Antelines holds the currently antecedent lines.

The general proof process consists of adding to Antelines using the available deduction rules until one of the subdeduction completion conditions is satisfied, at which point the top Goalstack entry is popped from there and added to Antelines, while all Antelines entries made after that goal was added to Goalstack are removed due to

boxing. PROOFMATRIX is updated along the way and printed out as the result once GOALSTACK is empty. Should Thinker reach a state where GOALSTACK is non-empty but no more inference steps are available to try, it will ask the user to help (by entering a new antecedent formula or requesting that a certain goal is shown), albeit in fully automated form the proof search would simply fail at that point.

A derivation of a formula F in Thinker begins with the line "show F", which means the derivation as a whole technically works just like the subdeductions described above. Upon introduction of a "show" line, it is first checked if any splitting strategies can be applied in order to immediately set new subgoals: $F_1 \leftrightarrow F_2$ can be shown by proving $F_1 \rightarrow F_2$ and $F_2 \rightarrow F_1$, and $F_1 \wedge F_2$ by proving F_1 and F_2. If a splitting rule applies, the new subgoals are proven sequentially and then joined with the appropriate introduction rule. Otherwise, an assumption of one of the three allowed types is introduced depending on the structure of F.

Whenever a new formula is introduced into the derivation, be it by assumption, by applying a rule, or by cancelling a "show", the function ONESTEP is called with the newly added formula to check if this formula can be used to prove the current goal in a single inference step. For example, if the formula F_3 is introduced while trying to prove $F_1 \rightarrow F_2$, ONESTEP would check ANTELINES for $F_3 \rightarrow F_2$, since this formula together with F_3 would allow applying the $\rightarrow$ elimination rule to derive F_2, which is one of the completion conditions for this goal. A similar function SIMPLEPROOF instead checks for a one-step inference of the given formula using the currently available information, but this function is only called in certain special cases, e.g. "show $F_1 \rightarrow F_2$"—if SIMPLEPROOF succeeds for F_2, we can complete this goal with the found one-step inference without even needing to introduce an assumption.

If ONESTEP fails, the next strategy is a mostly blind forward search that searches ANTELINES for formulas that make it possible to apply one of the many rules of inference. However, to avoid introducing additional complexity, only "simplifying" rules (i.e., elimination rules) are considered. As a last-ditch effort in case even this search cannot find any rules to apply, Thinker will attempt to show specific goals that would make further inference steps possible. For example, if ANTELINES contains the negation of an implication or disjunction, that implication or disjunction will be set as the new subgoal in order to cause a contradiction, which in turn would allow completion of the enveloping subdeduction.

While the most recent versions of Thinker (from the late 1980s) were written in C, its original version was written in Spitbol, a language focused on string pattern matching. Thus, formulas are stored as strings, with a dedicated function for splitting non-atomic formulas into their subformulas and determining the main connective. One of the main tasks in the proof process is searching ANTELINES for certain types of formulas, which is done with the help of *templates*—generalised formula strings which are used to look up all matching formulas in constant time. Looking up a specific formula in the string-indexed table, ANTELINES will provide information on where in the derivation that formula occurs, while looking up a template will provide a list of formulas matching it (which can then

be looked up in a second step). This is achieved by simply generating various matching templates for each new formula and storing the formula at those indices in the table. In addition to the main Antelines table, the program maintains circular linked lists comprised of formulas with the same main connective, which are used by the "blind" search strategies.

3.1.3 ANDP

The *Automated Natural Deduction Prover* (ANDP) [2, 3] is an "automated Gentzen system" created by Li Dafa and Jia Peifa. The derivations constructed by this program consist of goals $\Gamma \vdash F$, which are linked with proof rules to form a complete reasoning sequence for the final conclusion. The brief description given by Dafa and Peifa mainly covers the handling of quantifiers, but also includes certain strategic considerations that apply to propositional logic. The reasoning given for these strategies mainly draws from experimental evidence.

The order in which ANDP attempts to use inference rules is meant to achieve short derivations. The first step is bottom-up reasoning from the conclusion, where the rule to apply is simply chosen based on the form of the goal to prove:

- $\Gamma \vdash A \vee B$: The new goal (above the old one) is $\Gamma \vdash \neg A \to B$, via the implication rule.
- $\Gamma \vdash A \to B$: The new goal is $\Gamma, A \vdash B$, via the conditional rule (corresponding to $\to I$ of **N**).
- $\Gamma \vdash A \wedge B$: The two new goals are $\Gamma \vdash A$ and $\Gamma \vdash B$, via the conjunction rule (corresponding to $\wedge I$ of **N**).
- $\Gamma \vdash A \leftrightarrow B$: The new goal is $\Gamma \vdash (A \to B) \wedge (B \to A)$, via the equivalence rule.
- $\Gamma \vdash \neg A$: Attempt to push $\neg$ into A if possible, and make that the new goal.

Should this proof approach not succeed, the second step is to reason from top to bottom. Here, the available inference rules are tried in a fixed order, although different descriptions of the system disagree on the exact order, perhaps due to changes during development. What all sources agree on is that propositional rules such as modus ponens (the **N** rule $\to E$) or modus tollens are applied before quantifier rules, with the exception of case distinctions, which are tried last. Each time a rule succeeds and produces a new goal, the list of rules is again traversed from the start for that new goal.

The case-distinction rule, denoted by Cases, turns the disjuncts of a formula $A \vee B$ into new hypotheses (i.e., assumptions), much like $\vee E$ in **N**. This is done only as a last resort to avoid lengthening the derivation with too many unneeded formulas produced from these new assumptions, and even when Cases is applied, the disjunctions to which it is applied are strategically prioritised:

1. Apply Cases to initial assumptions (provided with the user's query).
2. Apply Cases to disjunctions from which it will not produce new constants.
3. Apply Cases to short formulas.

3.1.4 The ic-calculus Approach

Wilfried Sieg and John Byrnes [36] described a framework that allows searching for *normal* natural deduction derivations, in which the conclusions of introduction rules are never used as the major premisses of elimination rules (for a more detailed treatment of this concept, see also Chapter 4). Their approach is based on *intercalation calculi* (*ic-calculi*), which use the rules of natural deduction in certain restricted ways. While their paper does not mention any actual implementation of automated proof search, the design of the ic-calculi includes many strategies that would be useful for such a program.

An ic-calculus is defined by *ic-rules*, which operate on triples of the form $\alpha;\beta?G$ (*questions*), where α denotes the set of available assumptions, β is the set of formulas obtained from α via the elimination rules for $\wedge$ and $\rightarrow$, and G is the goal currently being worked towards. Since both α and β contain formulas available for use in further reasoning, their concatenation is also often relevant and denoted as $\alpha\beta$. The separation is due to negation rules and certain rules for predicate logic requiring premisses in α only.

Some strategies of the system can be encoded directly within the rules, as side conditions restricting their application. Three examples of rules with such conditions are given in the paper, although it is implied there could be more:

(i) $\alpha;\beta?G, \phi_1 \wedge \phi_2 \in \alpha\beta, \phi_i \notin \alpha\beta \Rightarrow \alpha;\beta,\phi_i?G$ (for $i \in \{1,2\}$) (corresponds to $\wedge E_1$, $\wedge E_2$ of **N**);

(ii) $\alpha;\beta?G, \phi_1 \vee \phi_2 \in \alpha\beta, \phi_1 \notin \alpha\beta, \phi_2 \notin \alpha\beta \Rightarrow \alpha \cup \{\phi_1\};\beta?G$ and $\alpha \cup \{\phi_2\};\beta?G$ (corresponds to $\vee E$ of **N**);

(iii) $\alpha;\beta?G, \phi_1 \rightarrow \phi_2 \in \alpha\beta, \phi_2 \notin \alpha\beta, \phi_1 \neq G \Rightarrow \alpha;\beta?\phi_1$ and $\alpha;\beta \cup \{\phi_2\}?G$ (corresponds to $\rightarrow E$ of **N**).

In all these cases, the side conditions simply prevent applying the rule in situations where it would produce no new information. For the same purpose of avoiding repetition, the two rules that model proof by contradiction (in **N** terms, $\neg I$ and *CRF*) can never both be applied to the same formula, as the $\neg I$ equivalent is only usable with non-negated formulas.

Applying ic-rules to an initial question $\alpha?G$ (with empty β omitted) yields an *ic-tree* whose branches represent *subquestions* for $\alpha?G$. The rules only allow for finitely many subquestions to be formulated, and by evaluating the subquestions it is possible to find either a normal derivation leading from α to G or a counterexample to the existence of such a derivation.

$$\frac{}{[A, \Gamma \vdash A \downarrow]}\ Id \qquad \frac{[\Gamma \vdash A \downarrow]}{[\Gamma \vdash A \Uparrow]}\ \downarrow\Uparrow \qquad \frac{[\neg A, \Gamma \vdash \bot \downarrow]}{[\Gamma \vdash A \Uparrow]}\ \bot E_{\mathrm{C}}$$

$$\frac{[\Gamma \vdash A \Uparrow] \quad [\Gamma \vdash B \Uparrow]}{[\Gamma \vdash A \wedge B \Uparrow]}\ \wedge I \qquad \frac{[\Gamma \vdash A_0 \wedge A_1 \downarrow]}{[\Gamma \vdash A_k \downarrow]}\ \wedge E_k \quad k \in \{0, 1\}$$

$$\frac{[\Gamma \vdash A_k \Uparrow]}{[\Gamma \vdash A_0 \vee A_1 \Uparrow]}\ \vee I_k \quad k \in \{0, 1\}$$

$$\frac{[\Gamma \vdash A \vee B \downarrow] \quad [A, \Gamma \vdash C \Uparrow] \quad [B, \Gamma \vdash C \Uparrow]}{[\Gamma \vdash C \Uparrow]}\ \vee E$$

$$\frac{[A, \Gamma \vdash B \Uparrow]}{[\Gamma \vdash A \to B \Uparrow]}\ \to I \qquad \frac{[\Gamma \vdash A \to B \downarrow] \quad [\Gamma \vdash A \Uparrow]}{[\Gamma \vdash B \downarrow]}\ \to E$$

Figure 3.1: The calculus **Nc**.

An ic-tree consists of two types of nodes: *Question nodes* and *rule nodes*, with the root being the initial question node $\alpha?G$. When processing a question node, it is first checked if $G \in \alpha\beta$, and if so, the current branch is closed with the label "Y". Otherwise, we add, as children of the question node, rule nodes for each rule that can be applied to the question without causing repetition, and the children of each rule node are the question nodes resulting from that rule. If a question node has no applicable rules, the branch is instead closed with the label "N". Each new question node is processed in this way until all branches are closed.

3.1.5 Ncr and Nbu

Mauro Ferrari and Camillo Fiorentini developed a natural deduction proof search procedure for classical propositional logic using a calculus **Ncr** [5]. They further refined this approach with the calculus **Nbu** and evaluated an actual implementation in their more recent work regarding intuitionistic propositional logic (IPL) [6], but we will first consider **Ncr** alone and then briefly summarise relevant improvments from **Nbu**.

The procedure follows the same basic principle as ic-calculi, applying bottom-up reasoning to conclusions and top-down reasoning to assumptions separately in order to build up partial derivations that can then be joined together to make a normal natural deduction derivation. This strategy is encoded in the calculus **Nc**, which operates on sequents of the form $\Gamma \vdash F \Uparrow$ or $\Gamma \vdash F \downarrow$, with the arrow labels denoting them as part of bottom-up and top-down reasoning, respectively. The rules of **Nc** are listed in Figure 3.1.

$$\frac{}{[\Gamma; H \vdash H \downarrow; \Delta;]}\ Id \qquad \frac{[\Gamma_H; H \vdash p \downarrow; p, \Delta; \Theta]}{[H, \Gamma \vdash p \Uparrow; \Delta]}\ \downarrow\Uparrow \qquad \frac{[\Gamma_H; H \vdash \bot \downarrow; F, \Delta; \Theta]}{[H, \Gamma \vdash F \Uparrow; \Delta]}\ \bot E_I$$

$$\frac{[\Gamma_H; H \vdash p \downarrow; F, p, \Delta; \Theta]}{[H, \Gamma \vdash F \Uparrow; p, \Delta]}\ R_p \qquad \frac{[\Gamma \vdash D \Uparrow; F, \Delta_D]}{[\Gamma \vdash F \Uparrow; D, \Delta]}\ R_c \qquad D \notin \mathcal{F}_\mathrm{p}$$

$$\frac{[\Gamma \vdash A \Uparrow; \Delta] \quad [\Gamma \vdash B \Uparrow; \Delta]}{[\Gamma \vdash A \wedge B \Uparrow; \Delta]}\ \wedge I \qquad \frac{[\Gamma; H \vdash A_0 \wedge A_1 \downarrow; \Delta; \Theta]}{[\Gamma; H \vdash A_k \downarrow; \Delta; A_{1-k}, \Theta]}\ \wedge E_k \qquad k \in \{0, 1\}$$

$$\frac{[\Gamma \vdash A \Uparrow; B, \Delta]}{[\Gamma \vdash A \vee B \Uparrow; \Delta]}\ \vee I$$

$$\frac{[\Gamma_H; H \vdash A \vee B \downarrow; F, \Delta; \Theta] \quad [A, \Gamma_H, \Theta \vdash F \Uparrow; \Delta] \quad [B, \Gamma_H, \Theta \vdash F \Uparrow; \Delta]}{[H, \Gamma \vdash F \Uparrow; \Delta]}\ \vee E$$

$$\frac{[A, \Gamma \vdash B \Uparrow; \Delta]}{[\Gamma \vdash A \to B \Uparrow; \Delta]}\ \to I \qquad \frac{[\Gamma; H \vdash A \to B \downarrow; \Delta; \Theta] \quad [\Gamma, \Theta \vdash A \Uparrow; \Delta]}{[\Gamma; H \vdash B \downarrow; \Delta; \Theta]}\ \to E$$

$$p \in \mathcal{V},\ F \in \mathcal{F}_\mathrm{p},\ \Lambda_A = \Lambda \setminus A$$

Figure 3.2: The calculus **Ncr**.

Of special note are the coercion rule $\downarrow\Uparrow$, the contradiction rule $\bot E_C$, the case distinction rule $\vee E$, and the implication elimination rule $\to E$. These rules mix the two directions of reasoning, with $\downarrow\Uparrow$ and $\bot E_C$ serving as the primary ways to join partial derivations. The rule $\vee E$ can be used similarly but requires supplemental bottom-up premisses, and $\to E$ instead introduces some unfortunate additional complexity in top-down reasoning by allowing it to cross back into the bottom-up direction. This is because even in a normal derivation, the minor premiss of an elimination rule can still be derived by an introduction rule, thus potentially requiring bottom-up reasoning.

This calculus is not well-suited for efficient proof search, as it can fall victim to loops caused by repetition of assumptions and contains many backtrack points. An approach to solve this involves tracking additional information in each sequent, replacing $\bot E_C$ with its intuitionistic counterpart $\bot E_I$, and introducing *restart rules* R_p and R_c to resume a previously "paused" proof search on a stored formula. The result is the considerably more complex calculus **Ncr** ("**Nc** with restart"), whose rules are listed in Figure 3.2.

Top-down sequents experience the most changes, now having a *head formula* H set apart from the rest of their assumptions and tracking both a *restart set* Δ and a *resource set* Θ. H is determined by the assumption from which the top-down derivation starts via

$$\frac{}{[A, \Gamma \Rightarrow A \downarrow]}\ Id \qquad \frac{[\Gamma \Rightarrow p \downarrow]}{[\Gamma \Rightarrow p \uparrow^l]}\ \downarrow\uparrow \quad p \in \mathcal{V} \qquad \frac{[\Gamma \Rightarrow \bot \downarrow]}{[\Gamma \Rightarrow F \uparrow^l]}\ \bot E \quad F \in \mathcal{V} \cup \{\bot\}$$

$$\frac{[\Gamma \Rightarrow A \uparrow^l] \quad [\Gamma \Rightarrow B \uparrow^l]}{[\Gamma \Rightarrow A \wedge B \uparrow^l]}\ \wedge I \qquad \frac{[\Gamma \Rightarrow A_0 \wedge A_1 \downarrow]}{[\Gamma \Rightarrow A_k \downarrow]}\ \wedge E_k \quad k \in \{0, 1\}$$

$$\frac{[\Gamma \Rightarrow A_k \uparrow^b]}{[\Gamma \Rightarrow A_0 \vee A_1 \uparrow^l]}\ \vee I_k \quad k \in \{0, 1\} \qquad \frac{[\Gamma \Rightarrow A \to B \downarrow] \quad [\Gamma \Rightarrow A \uparrow^b]}{[\Gamma \Rightarrow B \downarrow]} \to E$$

$$\frac{[\Gamma \Rightarrow A \vee B \downarrow] \quad [A, \Gamma \Rightarrow D \uparrow^u] \quad [B, \Gamma \Rightarrow D \uparrow^u]}{[\Gamma \Rightarrow D \uparrow^u]}\ \vee E \quad \begin{array}{l} D \in \mathcal{V} \cup \{\bot\} \text{ or } D = D_0 \vee D_1 \\ A \notin \Gamma \text{ and } B \notin \Gamma \end{array}$$

$$\frac{[\Gamma \Rightarrow B \uparrow^l]}{[\Gamma \Rightarrow A \to B \uparrow^l]} \to I_1\ A \in \Gamma \qquad \frac{[A, \Gamma \Rightarrow B \uparrow^u]}{[\Gamma \Rightarrow A \to B \uparrow^l]} \to I_2\ A \notin \Gamma$$

Figure 3.3: The calculus **Nbu**.

the *Id* rule and is intentionally made unavailable in certain areas of the derivation, Δ is used by $\bot E_I$ and the restart rules to pause and resume goals, and Θ stores unused conjunction branches discarded by a $\wedge E$ rule that are made available as assumptions in bottom-up premisses of top-down rules. The controlled use of resources facilitated by the head formula and the resource set follows the *LL-computation* paradigm [11]. Bottom-up sequents, however, are only extended with the restart set Δ. These adaptations prevent infinite loops.

Ncr is accompanied by a complementary calculus **RNcr**, which constructs derivations starting from *irreducible* sequents—precisely those to which no **Ncr** rules are applicable. This makes it possible to use a failing **Ncr** branch and turn it into a proof of refutability for the formula it was trying to prove, thus eliminating the need to backtrack from failing branches.

The calculus **Nbu** for intuitionistic propositional logic (see Figure 3.3) shares these properties, but is overall simpler due to not requiring restart rules or supplementary formula sets. Instead, its sequents feature labels b (blocked) and u (unblocked) which restrict the use of $\vee E$, and both $\vee E$ and $\to I$ (here split into $\to I_1$ and $\to I_2$) avoid loops via straightforward side conditions preventing repetition of assumptions.

Evaluation of a **Nbu** implementation using the Java framework JTabWb [7] showed performance similar to state-of-the-art IPL provers, though not outperforming them.

3.2 Strategies

Towards the description of strategies relevant for the ATP presented in this thesis, Hermes, we now consider which strategies are commonly used by the investigated systems, and which of them can also be applied to a refutational natural deduction system.

3.2.1 Strategies for Assertional Provers

The most significant overlap appears to be in the fact that the different proof search procedures all combine bottom-up and top-down reasoning in some way, usually using introduction rules for the former and elimination rules for the latter. In fact, this separation is central to the ic-calculi as well as to **Ncr** and **Nbu**, but even the older systems Oscar and ANDP make use of reasoning in both directions. And while Thinker builds its derivations entirely from top to bottom, the blind forward search procedure used, after more sophisticated strategies have failed, is intentionally limited to only using "simplifying" elimination rules, highlighting the fact that introduction rules are less suitable for top-down reasoning.

Given a distinction between elimination and introduction rules, it also appears reasonable to follow ic-calculi as well as **Ncr** and **Nbu** in their limitation to only producing normal derivations, since this does not restrict completeness in any way and reduces the complexity of proof search significantly by restricting the points where top-down and bottom-up derivations can join together. However, extending this idea to the refutational system will require us to establish that all $\overline{\mathbf{N}}$ derivations can be reduced to normal derivations, since, unlike for **N**, no such result exists yet in the literature.

While some systems do not make any mention of a specific order in which inference rules should be applied, suggesting it may not be a major factor for the ability of a natural deduction ATP to generate derivations in a timely fashion, there are still a few observations to be made in this regard. In Thinker, when a new goal is introduced, the program immediately attempts to split it into smaller subgoals, which intuitively makes sense because it is often easier to perform two simple derivations instead of one complex one. Once that is no longer possible, rules are applied based on their ability to reach the goal immediately, and beyond that the search becomes random. In ANDP, there is a fixed order in which rules are applied, with the major point of note that case distinctions are saved for last because of the additional complexity they may introduce. However, that may be related to the particular implementation of that system, since Oscar, ic-calculi, **Ncr**, and **Nbu** all contain case distinction rules without explicitly taking such precautions (although ic-calculi and **Nbu** instead feature a restriction where case distinctions cannot introduce duplicate assumptions).

3.2.2 Considerations for the Refutational Case

The complementary natural deduction calculus $\overline{\mathbf{N}}$ consists of rules that structurally resemble those of assertional natural deduction systems. We therefore expect the strategies

seen in the investigated provers to be applicable to this calculus as well, perhaps with some adjustments.

One thing that can be translated very straightforwardly is reasoning by cases, as it essentially implements the $\vee E$ rule of $\mathbf{N}$. The $\wedge E$ rule of $\overline{\mathbf{N}}$ follows the exact same scheme, so we simply have to switch the connective for which this type of reasoning is applied. The same can be done with the matching introduction rules, although they are not a significant factor in any strategies. On the other hand, the natural deduction rule $\wedge I$ is relevant to the splitting rules of Thinker, but can at first glance not simply be replaced by the two $\vee$ introduction rules found in $\overline{\mathbf{N}}$, since they each require a subdeduction. However, this difference is only actually relevant when trying to apply the rule outside the scope of all assumptions, because having derived both A and B under the same assumptions, the *Triv* rule can always be used to pull one of them into a nested subdeduction and so complete the required premisses. Even outside the scope of all assumptions, this will usually be possible, unless we specifically have one disjunct as a propositional constant (or its negation) derived by an atomic rule and the other disjunct as a formula containing the same propositional constant, such as in the example of $p \vee \neg p$.

Perhaps the most significant difference between the rules of $\mathbf{N}$ and $\overline{\mathbf{N}}$ is the handling of $\rightarrow$. Several of the investigated provers used conditionalisation as a major part of their strategy, but this is not possible in the complementary system, where $\rightarrow$ introduction is instead done in the same way as $\vee$ introduction.

On the other hand, the rules concerning $\neg$ only differ in terms of using $\top$ or $\bot$, which are not even present in any of the programs we looked at (they simply use the contradiction itself in place of the symbol). Thus, it is safe to say that the use of these rules will not significantly change in the refutational case.

Finally, unique to $\overline{\mathbf{N}}$ are the two atomic rules that allow freely to introduce a fresh propositional constant or its negation at any point. In top-down reasoning, these should be avoided at all costs, as they can produce an arbitrary number of formulas from absolutely no preconditions, but in bottom-up reasoning they can immediately provide us with certain required premisses. However, this scheme must be used with caution—the introduced propositional constants are not allowed to occur in any of the operative lines before the line where the rule is applied, so when reasoning backwards from the conclusion we risk locking ourselves out of formulas that would be necessary higher up in the derivation.

CHAPTER 4

Normal Derivations in N and $\overline{\mathbf{N}}$

One way we can significantly restrict the search space of a natural deduction system is by limiting ourselves to *normal derivations*, which obey the simple restriction that a formula derived by an introduction rule cannot be used as the major premiss of an elimination rule. The idea is that inferences of this form would always be redundant, since they can only produce information that must already have been present to apply the introduction rule in the first place.

As shown by Dag Prawitz [29], any derivation in an assertional natural deduction system for first-order logic that is restricted by omitting the $\vee$ connective and the $\exists$ quantifier can be converted to a normal derivation. Similar proofs for unrestricted systems were later provided by Richard Statman [38] and Gunnar Stålmarck [37].

In what follows, we will adopt the proof method employed by Stålmarck to show that derivations in **N** and a slightly modified variant of $\overline{\mathbf{N}}$ are reducible to normal derivations, meaning a restriction to only such derivations preserves completeness.

4.1 Proof Approach

First, we should more formally define the concept of a normal derivation for our purposes.

Definition 13. A *maximum formula* in a derivation in **N** or $\overline{\mathbf{N}}$ is a formula that occurs as both the conclusion of an introduction rule and the major premiss of an elimination rule. □

Definition 14. A derivation in **N** or $\overline{\mathbf{N}}$ is considered *normal* if it contains no maximum formula. □

The basic idea of Stålmarck's proof lies in reducing a given derivation to a normal derivation through repeated applications of *contractions*, i.e., the elimination of repeated

formulas. If each of these contractions is guaranteed to measurably reduce the number or degree of maximum formulas in the derivation, and any non-normal derivation has some applicable contraction, then it must be possible to reduce any derivation to one with zero maximum formulas. Formally, we assign each derivation δ a lexicographically ordered tuple $\langle n_\delta, m_\delta \rangle$, where n_δ is the smallest natural number so that no maximum formula in δ has a degree greater than n_δ, and m_δ is the number of maximum formulas with degree n_δ in the derivation. This tuple should become strictly smaller whenever we use a contraction to remove a maximum formula from δ.

More precisely, this approach proves the so-called (*weak*) *normalisation theorem*, which states that, for any given derivation, there is some sequence of reduction steps that ends in a normal derivation. The *strong normalisation theorem* further asserts that *every* sequence of reduction steps terminates in a normal derivation and has also been proven for natural deduction, but since we merely need the guarantee that normal derivations can represent all possible derivations, we will stick to the simpler proof of weak normalisation.

Further important notions are *simplified derivations* and *direct derivations*.

Definition 15. A derivation in **N** is *simplified* if each application of $\vee E$ has the conclusion $\bot$. Furthermore, a derivation in $\overline{\mathbf{N}}$ is *simplified* if each application of $\wedge E$ has the conclusion $\top$. □

Definition 16. A derivation in **N** is *direct* if the premiss of each application of *Triv* is an assumption. Furthermore, a derivation in $\overline{\mathbf{N}}$ is *direct* if the premiss of each application of *Triv* is either an assumption or the conclusion of an *At* or $\neg At$ application. □

All contractions will be applied to simplified, direct derviations and preserve these properties, which helps avoid major difficulties with case distinctions and the repetition of formulas.

4.2 The Assertional Case

Recall that the introduction rules of **N** are

$$CRF,\ \wedge I_1,\ \wedge I_2,\ \vee I_1,\ \vee I_2,\ \to I, \text{ and } \neg I,$$

while the elimination rules are

$$\wedge E_1,\ \wedge E_2,\ \vee E,\ \wedge E,\ \to E, \text{ and } \neg E.$$

Note that *CRV*, despite not explicitly being an introduction rule by name, must also be treated as one due to its ability to "hide" introduction rules within its subdeduction, enabling clearly non-normal derivations such as the following:

	$\vdots$	
i	A	given
	$\vdots$	
j	B	given
$j+1$	$\neg(A \wedge B)$	assumption
$j+2$	B	*Triv*
$j+3$	$A \wedge B$	$\wedge I_1$: i, $j+2$
$j+4$	$\bot$	$\neg E$: $j+1$, $j+3$
$j+5$	$A \wedge B$	*CRF*: $j+1$–$j+4$
$j+6$	A	$\wedge E_1$: $j+5$

Theorem 5. *Any derivation in* **N** *establishing* $\Gamma \vdash F$ *can be transformed to a normal derivation in* **N** *with the same assumptions and conclusion.*

For showing this theorem, as stated before, we will be working with simplified and direct derivations. So, first we need to adapt a given derivation so that it fulfills these properties. This can be achieved by means of contractions as discussed below.

- Consider the following derivation:

	$\vdots$	
i	$A \vee B$	given
$i+1$	A	assumption
	$\vdots$	
j	C	given
$j+1$	B	assumption
	$\vdots$	
k	C	given
$k+1$	C	$\vee E$: i, $i+1$–j, $j+1$–k

  If $C \neq \bot$, replace the above derivation with the derivation depicted in Figure 4.1.

- If F at line i is not an assumption, replace

	$\vdots$	
i	F	given
	$\vdots$	
j	F	*Triv*: i

  with

	$\vdots$	
i	F	given

  .

  While this contraction at first glance appears to be merely deleting the lines after i, note that the resulting derivation has to end on the same assumptions as the original one. Therefore, any assumptions made after the original line i must be moved before the new line i.

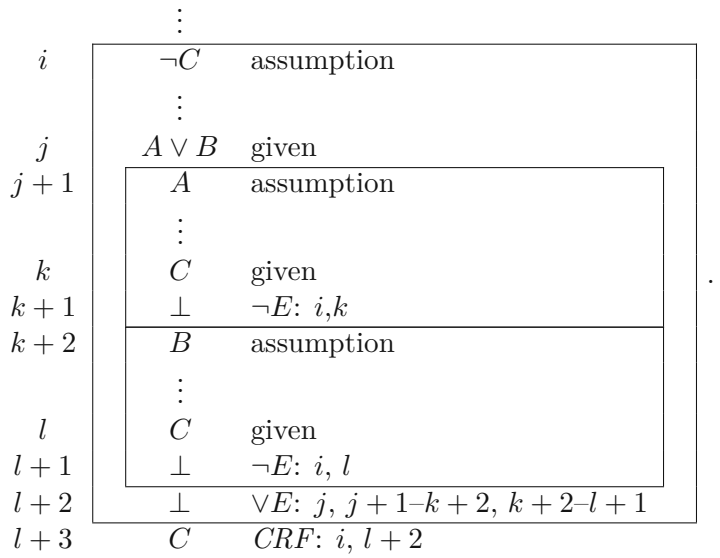


Figure 4.1: Derivation replacement for $\vee E$ and $C \neq \bot$.

Lemma 1. *Any derivation in* **N** *can be reduced to a derivation that is both simplified and direct.*

Proof. If the derivation is not simplified, it contains $n_1 > 0$ applications of $\vee E$ that have a conclusion other than $\bot$. Applying the first contraction listed above to remove such a line instead introduces a $\vee E$ application with conclusion $\bot$, and only $n_1 - 1$ applications of $\vee E$ with other conclusions (namely those that were not removed in this step) remain. Therefore, we will have a simplified derivation after n_1 applications of the contraction.

If the derivation is not direct, it contains $n_2 > 0$ applications of *Triv* whose premiss is not an assumption. Applying the second contraction listed above introduces no new lines to the derivation, so the number of such *Triv* applications must now be $n_2 - 1$, and n_2 applications of the contraction yield a direct derivation.

The contraction to remove $\vee E$ introduces no new instances of *Triv* with a non-assumption premiss, and the contraction to remove *Triv* introduces no new instances of $\vee E$, which means exhaustively applying both contractions in any order will ultimately yield a simplified, direct derivation. □

Next, we must define contractions for each case in which a maximum formula can occur, so that applying any contraction to an appropriate maximum formula in the derivation δ results in δ' with $\langle n_{\delta'}, m_{\delta'} \rangle < \langle n_\delta, m_\delta \rangle$. The simplest cases are those where a maximum formula is introduced by an introduction rule and used as the major premiss for exactly the corresponding elimination rule:

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \\ & \vdots & \\ j & B & \text{given} \\ j+1 & A \wedge B & \wedge I_1\text{: } i, j \\ j+2 & A & \wedge E_1\text{: } i \end{array}$$

with

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \end{array}$$

Obviously, contractions of the same form can be used if we have $\wedge I_2$ instead of $\wedge I_1$ and/or $\wedge E_2$ instead of $\wedge E_1$. Just like with the *Triv* contraction above, assumptions originally introduced between lines i and j must be moved so that the new derivation still ends on all the right assumptions.

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \\ i+1 & A \vee B & \vee I_1\text{: } i \\ i+2 & \boxed{\begin{array}{l} A \\ \vdots \\ \bot \end{array}} & \begin{array}{l} \text{assumption} \\ \\ \text{given} \end{array} \\ j+1 & \boxed{\begin{array}{l} B \\ \vdots \\ \bot \end{array}} & \begin{array}{l} \text{assumption} \\ \\ \text{given} \end{array} \\ k+1 & \bot & \vee E\text{: } i+1, i+2\text{–}j, j+1\text{–}k \end{array}$$

with

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \\ & \vdots & \\ j & \bot & \text{given} \end{array} .$$

For the case where $\wedge I_2$ is used to deduce the maximum formula, simply switch A for B instead.

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & A & \text{assumption} \\
 & \vdots & \\
j & B & \text{given} \\
j+1 & A \to B & \to I\text{: } i\text{–}j \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & B & \to E\text{: } j+1,\, k
\end{array}$$

with

$$\begin{array}{lll}
 & \vdots & \\
i & A & \text{given} \\
 & \vdots & \\
j & B & \text{given}
\end{array}$$

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & A & \text{assumption} \\
 & \vdots & \\
j & \bot & \text{given} \\
j+1 & \neg A & \neg I\text{: } i\text{–}j \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & \bot & \neg E\text{: } j+1,\, k
\end{array}$$

with

$$\begin{array}{lll}
 & \vdots & \\
i & A & \text{given} \\
 & \vdots & \\
j & \bot & \text{given}
\end{array}$$

In each of the previous contractions, it is immediately obvious that they outright remove a maximum formula, as all lines of the resulting derivation already occur in the original derivation and the line with the maximum formula is not among them. Thus, we have

either $n_{\delta'} < n_\delta$ (in case it was the last maximum formula of degree n_δ) or $n_{\delta'} = n_\delta$ and $m_{\delta'} < m_\delta$—either way, it holds that $\langle n_{\delta'}, m_{\delta'} \rangle < \langle n_\delta, m_\delta \rangle$ by lexicographical ordering.

Things become a bit less simple once we also look at the contractions meant to remove maximum formulas derived by *CRF*.

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & \boxed{\begin{array}{ll} \neg(A \wedge B) & \text{assumption} \\ \vdots & \\ \bot & \text{given} \end{array}} & \\
j+1 & A \wedge B & \textit{CRF}\text{: } i\text{–}j \\
j+2 & A & \wedge E_1\text{: } j+1
\end{array}$$

  with

$$\begin{array}{lll}
 & \vdots & \\
i & \boxed{\begin{array}{ll} \neg A & \text{assumption} \\ \boxed{\begin{array}{ll} A \wedge B & \text{assumption} \\ A & \wedge E_1\text{: } i+1 \\ \bot & \neg E\text{: } i, i+2 \end{array}} & \\ \neg(A \wedge B) & \neg I\text{: } i+1\text{–}i+3 \\ \vdots & \\ \bot & \text{given} \end{array}} & \\
j+1 & A & \textit{CRF}\text{: } i\text{–}j
\end{array}$$

  Once again, the case for $\wedge E_2$ merely uses B in place of A.

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & \boxed{\begin{array}{ll} \neg(A \vee B) & \text{assumption} \\ \vdots & \\ \bot & \text{given} \end{array}} & \\
j+1 & A \vee B & \textit{CRF}\text{: } i\text{–}j \\
j+2 & \boxed{\begin{array}{ll} A & \text{assumption} \\ \vdots & \\ \bot & \text{given} \end{array}} & \\
k+1 & \boxed{\begin{array}{ll} B & \text{assumption} \\ \vdots & \\ \bot & \text{given} \end{array}} & \\
l+1 & \bot & \vee E\text{: } j+1,\ j+2\text{–}k,\ k+1\text{–}l
\end{array}$$

with

$$
\begin{array}{lcl}
 & \vdots & \\
i & A \vee B & \text{assumption} \\
i+1 & \boxed{\begin{array}{lcl} A & \text{assumption} \\ \vdots & & \\ j & \bot & \text{given} \end{array}} & \\
j+1 & \boxed{\begin{array}{lcl} B & \text{assumption} \\ \vdots & & \\ k & \bot & \text{given} \end{array}} & \\
k+1 & \bot & \vee E\text{: } i,\ i+1\text{–}j,\ j+1\text{–}k \\
k+2 & \neg(A \vee B) & \neg I\text{: } i\text{–}k+1 \\
 & \vdots & \\
l & \bot & \text{given}
\end{array}
$$

- Replace a derivation of the form

$$
\begin{array}{lcl}
 & \vdots & \\
i & \boxed{\begin{array}{lcl} \neg(A \to B) & \text{assumption} \\ \vdots & & \\ j & \bot & \text{given} \end{array}} & \\
j+1 & A \to B & \mathit{CRF}\text{: } i\text{–}j \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & B & \to E : j+1,\ k
\end{array}
$$

with

$$
\begin{array}{lcl}
 & \vdots & \\
i & \boxed{\begin{array}{lcl} \neg B & \text{assumption} \\ A \to B & \text{assumption} \\ \vdots & & \\ j & A & \text{given} \\ j+1 & B & \to E\text{: } i+1,\ j \\ j+2 & \bot & \neg E\text{: } i,\ j+1 \end{array}} & \\
j+3 & \neg(A \to B) & \neg I\text{: } i+1\text{–}j+2 \\
 & \vdots & \\
k & \bot & \text{given} \\
k+1 & B & \mathit{CRF}\text{: } i\text{–}k
\end{array}
$$

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & \boxed{\neg\neg A} & \text{assumption} \\
 & \vdots & \\
j & \bot & \text{given} \\
j+1 & \neg A & \mathit{CRF}\text{: } i\text{–}j \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & \bot & \neg E : j+1,\, k
\end{array}$$

  with

$$\begin{array}{lll}
 & \vdots & \\
i & \boxed{\neg A} & \text{assumption} \\
 & \vdots & \\
j & A & \text{given} \\
j+1 & \bot & \neg E\text{: } i,\, j \\
i+2 & \neg\neg A & \neg I\text{: } i\text{–}j+1 \\
 & \vdots & \\
k & \bot & \text{given}
\end{array}$$

The common problem these contractions run into is that, while the maximum formula under consideration is indeed removed, we also introduce a new instance of the introduction rule $\neg I$. If the conclusion of this rule is then used as the major premiss of a $\neg E$ application, that would make it a new maximum formula in δ' that was not present in δ. Worse yet, due to the added $\neg$ connective, this new formula has a higher degree than the one we removed, so we would actually end up with $\langle n_{\delta'}, m_{\delta'}\rangle > \langle n_\delta, m_\delta\rangle$—the opposite of our aim.

Therefore, in addition to the *proper contractions* that simply remove maximum formulas, we require an additional group of *assumption contractions* to preemptively clean up derivations that would lead to the described scenario:

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & \boxed{\neg(A \wedge B)} & \text{assumption} \\
 & \vdots & \\
j & A \wedge B & \text{given} \\
j+1 & \bot & \neg E\text{: } i,\, j \\
 & \vdots & \\
k & \bot & \text{given} \\
k+1 & A \wedge B & \mathit{CRF}\text{: } i\text{–}k \\
k+2 & A & \wedge E_1\text{: } k+1
\end{array}$$

with

$$
\begin{array}{rcl}
 & \vdots & \\
i & \neg A & \text{assumption} \\
i+1 & \neg(A \wedge B) & \text{assumption} \\
 & \vdots & \\
j & A \wedge B & \text{given} \\
j+1 & A & \wedge E_1\colon j \\
j+2 & \bot & \neg E\colon i,\ j+1 \\
 & \vdots\!\vdots & \\
k & \bot & \text{given} \\
k+1 & A \wedge B & \mathit{CRF}\colon i+1\text{–}k \\
k+2 & A & \wedge E_1\colon k+1 \\
k+3 & \bot & \neg E\colon i,\ k+2 \\
k+4 & A & \mathit{CRF}\colon i\text{–}k+3
\end{array}
\quad .
$$

Here, the symbol

$$\vdots\!\vdots$$

is used to indicate an arbitrary sequence of derivation steps in which one *may* discharge previously introduced assumptions, in particular any assumptions introduced within the lines $i+2$ to $j-1$.

To see the necessity of this, consider a derivation establishing $\{B, \neg A \to \bot\} \vdash A$:

$$
\begin{array}{rcl}
1. & \neg A \to \bot & \text{assumption} \\
2. & B & \text{assumption} \\
3. & \neg(A \wedge B) & \text{assumption} \\
4. & A & \text{assumption} \\
5. & A \wedge B & \wedge I_2\colon 2,\ 4 \\
6. & \bot & \neg E\colon 3,\ 5 \\
7. & \neg A & \neg I\colon 4\text{–}6 \\
8. & \bot & \to E\colon 1,\ 7 \\
9. & A \wedge B & \mathit{CRF}\colon 3\text{–}8 \\
10. & A & \wedge E_1\colon 9
\end{array}
$$

Since the assumption $\neg(A \wedge B)$ would be replaced by the conclusion of an $\neg I$ application when using the proper contraction, and thus become a new maximum formula, derivations of this form must be adapted as well.

- Replace a derivation of the form

	$\vdots$	
i	$\neg(A \vee B)$	assumption
	$\vdots$	
j	$A \vee B$	given
$j+1$	$\bot$	$\neg E$: i, j
	$\vdots\!\vdots$	
k	$\bot$	given
$k+1$	$A \vee B$	CRF: i–k
$k+2$	A	assumption
	$\vdots$	
l	$\bot$	given
$l+1$	B	assumption
	$\vdots$	
m	$\bot$	given
$m+1$	$\bot$	$\vee E$: $k+1$, $k+2$–l, $l+1$–m

with

	$\vdots$	
i	$\neg(A \vee B)$	assumption
	$\vdots$	
j	$A \vee B$	given
$j+1$	A	assumption
	$\vdots$	
k	$\bot$	given
$k+1$	B	assumption
	$\vdots$	
l	$\bot$	given
$l+1$	$\bot$	$\vee E$: j, $j+1$–k, $k+1$–l
	$\vdots\!\vdots$	
m	$\bot$	given
$m+1$	$A \vee B$	CRF: i–m
$m+2$	A	assumption
	$\vdots$	
n	$\bot$	given
$n+1$	B	assumption
	$\vdots$	
o	$\bot$	given
$o+1$	$\bot$	$\vee E$: $m+1$, $m+2$–n, $n+1$–o

- Replace a derivation of the form

$$
\begin{array}{lll}
 & \vdots & \\
i & \neg\neg A & \text{assumption} \\
 & \vdots & \\
j & \neg A & \text{given} \\
j+1 & \bot & \neg E\text{: } i, j \\
 & \vdots & \\
k & \bot & \text{given} \\
k+1 & \neg A & \mathit{CRF}\text{: } i+1\text{–}k \\
 & \vdots & \\
l & A & \text{given} \\
l+1 & \bot & \neg E\text{: } k+1, l
\end{array}
$$

with

$$
\begin{array}{lll}
 & \vdots & \\
i & \neg\neg A & \text{assumption} \\
 & \vdots & \\
j & \neg A & \text{given} \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & \bot & \neg E\text{: } j, k \\
 & \vdots & \\
l & \bot & \text{given} \\
l+1 & \neg A & \mathit{CRF}\text{: } i\text{–}l \\
 & \vdots & \\
m & A & \text{given} \\
m+1 & \bot & \neg E\text{: } l+1, m
\end{array}
$$

- Replace a derivation of the form

$$
\begin{array}{lll}
 & \vdots & \\
i & \neg(A \to B) & \text{assumption} \\
 & \vdots & \\
j & A \to B & \text{given} \\
j+1 & \bot & \neg E\text{: } i, j \\
 & \vdots & \\
k & \bot & \text{given} \\
k+1 & A \to B & \mathit{CRF}\text{: } i\text{–}k \\
 & \vdots & \\
l & A & \text{given} \\
l+1 & B & \to E\text{: } k+1, l
\end{array}
$$

with

$$
\begin{array}{rcl}
 & \vdots & \\
i & \neg B & \text{assumption} \\
i+1 & \neg(A \to B) & \text{assumption} \\
 & \vdots & \\
j & A \to B & \text{given} \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & B & \to E\text{: } j,\ k \\
k+2 & \bot & \neg E\text{: } i,\ k+1 \\
 & \vdots & \\
l & \bot & \text{given} \\
l+1 & A \to B & \mathit{CRF}\text{: } i+1\text{–}l \\
 & \vdots & \\
m & A & \text{given} \\
m+1 & B & \to E\text{: } l+1,\ m \\
m+2 & \bot & \neg E\text{: } i,\ m+1 \\
m+3 & B & \mathit{CRF}\text{: } i\text{–}m+2
\end{array}
$$

These contractions do not immediately reduce the number of maximum formulas in the derivation, and may in fact increase it via the additional elimination rule application in the subdeduction. However, they cannot themselves introduce a maximum formula with higher degree and let us avoid the cases where subsequent contractions would do so, which is still a meaningful progress towards normalisation.

Having defined all the contractions, we can see immediately that applying them to a simplified derivation will indeed always produce a simplified derivation, because none of them introduce any lines in which $\vee E$ is used to conclude a formula other than $\bot$. Also, applying them to a direct derivation will always produce a direct derivation because they introduce no applications of *Triv* whose premiss is not an assumption.

What remains to be shown is that these contractions are sufficient to transform any simplified and direct derivation into a normal derivation, which we will do in two steps.

Lemma 2. *If δ is a simplified and direct derivation in* **N** *establishing $\Gamma \vdash F$, there exists a simplified and direct* **N** *derivation δ' with the same assumptions and conclusion in which no assumption contractions are applicable to maximum formulas of degree $n_{\delta'}$, and $n_{\delta'} = n_\delta$.*

Proof. For an assumption contraction to be applicable to a maximum formula F, it must be concluded by *CRF*, and the associated subdeduction with assumption $\neg F$ must contain an application of $\neg E$ having that assumption as its major premiss. Each contraction does remove such a $\neg E$ application, but we cannot immediately say that this always

reduces the number of formulas in the derivation to which assumption contractions can be applied, because the additional copy of F that possibly becomes a new maximum formula may itself have been derived in a way that then demands an assumption contraction. However, a derivation of finite length obviously cannot infinitely nest *CRF* applications, and so reapplying the same contraction to the new maximum formula will eventually give us a derivation with fewer assumption contractions left to apply.

We can therefore simply use the following approach: Take a maximum formula of degree n_δ to which an assumption contraction is applicable in δ, apply that contraction, and if necessary apply it to the copy of F inside the subdeduction, repeating the process until that formula is no longer a maximum formula that an assumption contraction can be applied to. This gives us a new derivation δ^1 with $n_{\delta^1} = n_\delta$ and $m_{\delta^1} \geq m_\delta$. Doing it again with the next maximum formula of degree n_δ to which an assumption contraction is applicable gives us δ^2, and so on for a sequence of derivations $\delta, \delta^1, \delta^2, \ldots, \delta^k$. Our previous observation tells us that this sequence is finite, $n_{\delta^k} = n_\delta$, and δ^k by definition contains no maximum formulas of degree n_{δ^k} to which an assumption contraction can be applied, making it exactly the δ' we are after. □

Lemma 3. *Any simplified and direct derivation in* **N** *establishing* $\Gamma \vdash F$ *can be reduced to a normal derivation showing* $\Gamma \vdash F$*.*

Proof. Let δ_0 be the initial derivation. Then, δ_0 contains exactly m_{δ_0} maximum formulas of degree n_{δ_0}, and no maximum formulas with a degree greater than that. By Lemma 2, we can obtain a derivation δ_0' to which no assumption contractions are applicable, which rules out all cases where applying a proper contraction to a maximum formula of degree $n_{\delta_0'}$ $(= n_{\delta_0})$ would produce one of higher degree.

Thus, we can now simply take a maximum formula of degree $n_{\delta_0'}$ in δ_0' and apply the applicable proper contraction to it, resulting in a derivation δ_1 that has either $n_{\delta_1} < n_{\delta_0'}$, or $n_{\delta_1} = n_{\delta_0'}$ and $m_{\delta_1} < m_{\delta_0'}$. In the first case, we again exhaustively apply the assumption contractions to get δ_1' and then continue with the next proper contraction to get δ_2. In the second case, we simply proceed to δ_2 in the same way immediately, as there will still be no applicable assumption contractions.

Repeating this process until no maximum formulas are left gives us a finite sequence of derivations $\delta_0, \delta_1, \ldots, \delta_k$ where $\langle n_{\delta_{i+1}}, m_{\delta_{i+1}} \rangle < \langle n_{\delta_i}, m_{\delta_i} \rangle$. For the last element δ_k in particular, the tuple is $\langle 0, 0 \rangle$, meaning it contains no maximum formulas and is therefore a normal derivation. □

Now, by Lemma 1, any given derivation can be transformed to a simplified and direct derivation, and by Lemma 3, that simplified and direct derivation can be transformed to a normal derivation. This proves Theorem 5.

4.3 The Refutational Case

Trying to prove normalisation for the complementary calculus $\overline{\mathbf{N}}$ would be a fruitless effort for the simple reason that the opposite can very easily be demonstrated. Due to

the special requirements of the rules At and $\neg At$, as well as the fact that we cannot use premisses from outside the scope of all assumptions while an assumption is active, some derivations can only be made using a particular construction where a conclusion of $\vee I_1$ becomes the major premiss of $\vee E_2$, as seen in the example below.

1.	$\bot$	$\bot I$
2.	$\bot$	assumption
3.	$\neg p_0$	$\neg At$
4.	$\neg p_1$	$\neg At$
5.	$p_0 \wedge p_1$	assumption
6.	p_0	assumption
7.	$\top$	$\neg E$: 3, 6
8.	p_1	assumption
9.	$\top$	$\neg E$: 4, 8
10.	$\top$	$\wedge E$: 5, 6–7, 8–9
11.	$\neg(p_0 \wedge p_1)$	$\neg I$: 4–10
12.	$\bot \vee \neg(p_0 \wedge p_1)$	$\vee I_1$: 1, 2–11
13.	$\neg(p_0 \wedge p_1)$	$\vee E_2$: 12

The usual normalisation approach here would be to simply use the derivation of $\neg(p_0 \wedge p_1)$ that must exist within the premisses of $\vee I_1$, but this derivation specifically requires the presence of a "neutral" assumption like $\bot$. Deriving $\neg p_0$ and $\neg p_1$ before assumptions are made would mean they are not operative within the subderivations of $\wedge E$ where we need them, and the assumption $p_0 \wedge p_1$ contains both propositional constants and thus blocks the atomic rules for them once it is active.

However, as this is the only scenario that does not fit the requirements of normal derivations, we can instead consider the calculus $\overline{\mathbf{N}}_{BOX}$, which is simply $\overline{\mathbf{N}}$ extended by the following rule with the side condition that line n is not in the scope of any assumptions:

$$\dfrac{\begin{array}{|cc} & \vdots \\ m & \bot \\ & \vdots \\ n-1 & F \end{array}}{n \quad F}\ BOX$$

The soundness of this new rule is obvious from the fact that it is simply a shortcut for the specific combination of $\vee I_1$ and $\vee E_2$ we are trying to avoid. We consider BOX to be an introduction rule, i.e., its conclusion may not be the major premiss of an elimination rule in a normal derivation.

In summary, $\overline{\mathbf{N}}_{BOX}$ has the introduction rules

$$BOX,\ CRV,\ \wedge I_1,\ \wedge I_2,\ \vee I_1,\ \vee I_2,\ \to I_1,\ \to I_2,\ \text{and}\ \neg I,$$

and the elimination rules

$$\wedge E,\ \vee E_1,\ \vee E_2,\ \to E_1,\ \to E_2,\ \neg E.$$

We do not count the atomic rules as introduction rules despite their role of introducing new propositional constants, since they mostly do not interact with elimination rules.

Theorem 6. *Any derivation in $\overline{\mathbf{N}}_{BOX}$ establishing $\Gamma \dashv F$ can be transformed to a normal derivation in $\overline{\mathbf{N}}_{BOX}$ with the same assumptions and conclusion.*

Compared to the proof of the assertional case, the contractions here have some additional complications due to the restriction of the refutational system against reusing formulas from outside the scope of all assumptions once an assumption has been made. This already becomes apparent in the contractions to make derivations simplified and direct:

- If in the derivation below $C \neq \top$ and line $k+1$ is in the scope of some assumption, replace

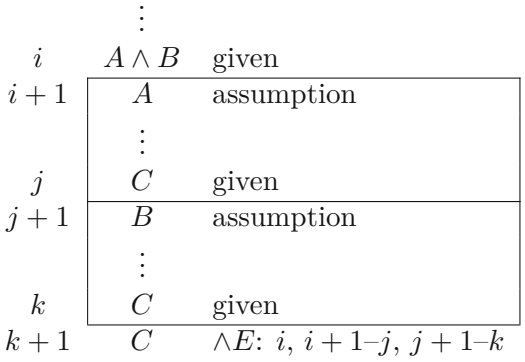
$$\begin{array}{rcl}
 & \vdots & \\
i & A \wedge B & \text{given} \\
i+1 & A & \text{assumption} \\
 & \vdots & \\
j & C & \text{given} \\
j+1 & B & \text{assumption} \\
 & \vdots & \\
k & C & \text{given} \\
k+1 & C & \wedge E\text{: } i,\ i+1\text{–}j,\ j+1\text{–}k
\end{array}$$

with

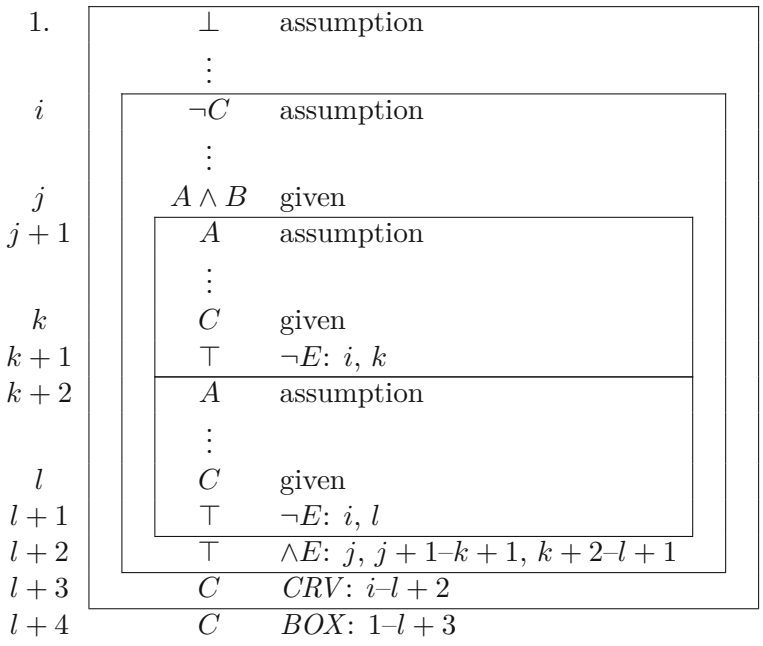
$$\begin{array}{rcl}
 & \vdots & \\
i & \neg C & \text{assumption} \\
 & \vdots & \\
j & A \wedge B & \text{given} \\
j+1 & A & \text{assumption} \\
 & \vdots & \\
k & C & \text{given} \\
k+1 & \top & \neg E\text{: } i,\ k \\
k+2 & A & \text{assumption} \\
 & \vdots & \\
l & C & \text{given} \\
l+1 & \top & \neg E\text{: } i,\ l \\
l+2 & \top & \wedge E\text{: } j,\ j+1\text{–}k+1,\ k+2\text{–}l+1 \\
l+3 & C & CRV\text{: } i\text{–}l+2
\end{array}$$

Any applications of atomic rules after line i that conflict with the assumption $\neg C$ must be moved so they stand before that assumption, and *Triv* must be used to recover them at the lines where they are needed.

- If in the derivation below $C \neq \top$ and line $k+1$ is outside the scope of all assumptions, replace



with



Again, problematic applications of atomic rules after line i have to be moved up, this time into the scope of the assumption $\bot$ alone, which cannot be in conflict with them since it contains no propositional constants.

- If F at line i is neither an assumption nor derived by At or $\neg At$, replace

$$\begin{array}{lll} & \vdots & \\ i & F & \text{given} \\ & \vdots & \\ j & F & \textit{Triv}\colon i \end{array}$$

with

$$\begin{array}{lll} & \vdots & \\ i & F & \text{given} \end{array}$$

Since the assumptions at the new line i must be the same as at the original line j, some atomic rule applications may need to move upwards here as well.

In addition to taking special care of atomic rule applications that would move into the scope of additional assumptions, the second contraction for $\wedge E$ requires placing a derivation from outside the scope of all assumptions into the scope of an assumption $\bot$, making use of the newly added BOX rule. This drastic change in context is, in fact, possible without violating the side conditions of the atomic rules.

Lemma 4. *For any derivation in $\overline{\mathbf{N}}_{BOX}$ establishing $\dashv F$, there also exists a derivation in $\overline{\mathbf{N}}_{BOX}$ showing $\{\bot\} \dashv F$, and the operative formulas at the last line of that derivation contain no propositional constants other than those in F.*

Proof. Having a derivation with no assumptions is somewhat helpful, as it means F is derived without relying on "unseen" information, limiting the shapes of possible inferences. We also do not need to think much about the case where F is derived by BOX, since the direct premiss of that rule is itself a subderivation $\{\bot\} \dashv F$.

For a closer inspection of the remaining cases, we proceed by induction on $\deg(F)$.

Base Case: $\deg(F) = 0$. By soundness of $\overline{\mathbf{N}}_{BOX}$, F cannot be $\top$, so one of the following cases applies:

- $F = \bot$, making it always derivable by $\bot I$, and also directly via the assumption $\bot$.
- F is a propositional constant that could have been derived by At or by CRV. In the former case, we can certainly still apply the At rule this way after adding the assumption $\bot$, because it does not contain any propositional constants. In the latter case, the derivation consists of a single subderivation with assumption $\neg F$, which can be moved into the scope of $\bot$ without conflicts because the formulas within it were operative to begin with. The operative formulas at the end of the new derivation are only the assumption $\bot$ and F itself, so clearly the restriction on operative propositional constants holds as well.

Induction Hypothesis: If $0 < \deg(F) \leq n-1$ and there is a derivation in $\overline{\mathbf{N}}_{BOX}$ showing $\dashv F$, there is also a derivation in $\overline{\mathbf{N}}_{BOX}$ showing $\{\bot\} \dashv F$, and the operative formulas at its last line contain no propositional constants other than those in F.

Induction Step: We distinguish by the outermost connective of F.

- If F has the form $F_1 \wedge F_2$, it is introduced by $\wedge I_1$, $\wedge I_2$, or CRV. If CRV, the reasoning given in the base case is applicable regardless of what F looks like, so we will disregard that case from here out. That leaves us with the $\wedge I$ rules, which have as their premiss either F_1 or F_2. As both of these formulas are subformulas of F, the needed one can be derived in the scope of $\bot$ by the induction hypothesis, and so $\{\bot\} \dashv F_1 \wedge F_2$ follows immediately, without any additional propositional constants in the operative formulas.
- If F has the form $F_1 \vee F_2$, it is derived by a $\vee I$ rule. For $\vee I_1$ in particular, that means the premisses are F_1 and a subderivation yielding F_2 under the assumption F_1. We have $\{\bot\} \dashv F_1$ by the induction hypothesis because F_1 is a subformula of F, and embedding the subderivation obviously does not pose any problems since the formulas it contains were already operative within itself before, and are not operative after discharging the assumption F_1. For $\vee I_2$, one only needs to switch the formulas.
- If F has the form $F_1 \to F_2$, it is derived by a $\to I$ rule. As those rules are shaped just like the $\vee I$ rules, except for the premiss F_1 being replaced by $\neg F_1$, we can generally apply the same reasoning again. However, matters are complicated by the fact that $\neg F_1$ is a formula of the same degree as $F_1 \to F_2$ in the particular case where F_2 is a propositional constant, $\bot$, or $\top$, so we do not always have guaranteed directly by the induction hypothesis that there is an appropriate derivation of it. But outside the scope of all assumptions, the final rule used to derive $\neg F_1$ in the original derivation must be CRV, $\neg I$, or $\neg At$, whose premisses do not feature any formulas outside of subderivations, meaning there is no room for conflicts when moved into the scope of $\bot$ and the only formula made operative in the end is $\neg F_1$.
- If F has the form $\neg F_1$, it is derived by $\neg I$ or by $\neg At$. Like CRV, the only premiss of $\neg I$ is a subderivation, so there are no newly operative formulas to worry about. Moreover, $\neg At$, like At, requires no premisses and its side conditon cannot conflict with the assumption $\bot$.

□

Lemma 5. *Any derivation in $\overline{\mathbf{N}}_{BOX}$ can be reduced to a derivation that is both simplified and direct.*

Proof. While the contractions are somewhat more complicated, the overall reduction process does not significantly differ from the assertional case. Each use of a contraction to

remove an instance of $\wedge E$ with conclusion other than $\top$ from a non-simplified derivation reduces the number of such $\wedge E$ instances, and each use of a contraction to remove an instance of *Triv* with a premiss other than an assumption or atomic rule from an indirect derivation reduces the number of such *Triv* instances. Furthermore, while the contractions might introduce new instances of *Triv* to properly place formulas introduced by atomic rules, these instances will always have as their premiss a formula introduced by At or $\neg At$, meaning they do not interfere with directness.

It follows that exhaustively applying the contractions will produce a simplified and direct $\overline{\mathbf{N}}_{BOX}$ derivation. □

The proper contractions that remove combinations of matching introduction and elimination rules are described in what follows.

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \\ i+1 & \boxed{\begin{array}{l} A \\ \vdots \\ B \end{array}} & \begin{array}{l} \text{assumption} \\ \\ \\ \text{given} \end{array} \\ j & & \\ j+1 & A \vee B & \vee I_1\colon\ i,\ i+1\text{–}j \\ j+2 & A & \vee E_1\colon\ j+1 \end{array}$$

  with

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \end{array}$$

  The same contraction holds for the pair of $\vee I_2$ and $\vee E_2$, but the case where introduction and elimination rules have different numbers (i.e., the formula at line i is not the same as the final conclusion) demands a different approach.

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & B & \text{given} \\ i+1 & \boxed{\begin{array}{l} B \\ \vdots \\ A \end{array}} & \begin{array}{l} \text{assumption} \\ \\ \\ \text{given} \end{array} \\ j & & \\ j+1 & A \vee B & \vee I_2\colon\ i,\ i+1\text{–}j \\ j+2 & A & \vee E_1\colon\ j+1 \end{array}$$

with

$$\begin{array}{lll} & \vdots & \\ i & B & \text{given} \\ & \vdots & \\ i & A & \text{given} \end{array}$$

A pitfall to keep in mind with this and other contractions is that removing the derivation of A from the scope of B may cause problems, even if we have B without the need for an assumption. Specifically, if the derivation is now outside the scope of all assumptions, rules applied within subderivations in that derivation may lose access to their premisses, because a formula derived without assumptions is only attainable as long as no assumptions are made.

In such cases, the *BOX* rule comes to our rescue and allows us to rewrite the derivation accordingly:

$$\begin{array}{lll} 1. & \boxed{\begin{array}{lll} \bot & \text{assumption} \\ \vdots & & \\ B & \text{given} \\ \vdots & & \\ A & \text{given} \end{array}} & \\ j+1 & A & \mathit{BOX}\colon 1\text{–}j \end{array}$$

By Lemma 4, moving the derivation of B into the scope of $\bot$ is not only possible, but also makes no formulas operative that contain propositional constants other than those in B. This means any applications of the atomic rules in the derivation of A are safe to embed, since they had to respect the assumption B in the original derivation already, and we can apply the contraction with or without surrounding assumptions.

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \\ i+1 & A \wedge B & \wedge I_1\colon i \\ i+2 & \boxed{\begin{array}{lll} A & \text{assumption} \\ \vdots & & \\ \top & \text{given} \end{array}} & \\ j+1 & \boxed{\begin{array}{lll} B & \text{assumption} \\ \vdots & & \\ \top & \text{given} \end{array}} & \\ k+1 & \top & \wedge E\colon i+1,\ i+2\text{–}j,\ j+1\text{–}k \end{array}$$

with

$$\begin{array}{cll} & \vdots & \\ i & A & \text{given} \\ & \vdots & \\ j & \top & \text{given} \end{array}$$

Obviously, we do not need to worry about the case where a derivation of $\top$ occurs outside the scope of all assumptions, and thus there is no boxed version of this contraction. To handle $A \wedge B$ introduced by $\wedge I_2$, one must simply use the derivation found in the other subderivation.

- Replace a derivation of the form

$$\begin{array}{cll} & \vdots & \\ i & \neg A & \text{given} \\ i+1 & \boxed{\begin{array}{l}\neg A \\ \vdots \\ B\end{array}} & \begin{array}{l}\text{assumption} \\ \\ \text{given}\end{array} \\ j+1 & A \to B & \to I_1\colon\ i,\ i+1\text{–}j \\ j+2 & \neg A & \to E_1\colon\ j+1 \end{array}$$

with

$$\begin{array}{cll} & \vdots & \\ i & \neg A & \text{given} \end{array}$$

While the same approach also works for $\to E_2$, $\to E_1$ in particular causes some trouble for our proof because, unlike all other elimination rules considered so far, its conclusion $\neg A$ is potentially of the same degree as $A \to B$. Therefore, if it is derived by an introduction rule on line i and later in the derivation used as the major premiss of $\neg E$, we have not actually succeeded in reducing the overall degree of maximum formulas by applying this contraction. However, this new maximum formula would be removed with a different contraction (as its outermost connective is $\neg$, not $\to$) and that contraction does result in a lower-degree maximum formula (if any), so we do still make progress.

To get around this, we can simply say that in the specific case where applying this contraction produces a new maximum formula of the same degree, that formula is immediately (i.e., within the same reduction step) removed by the appropriate contraction, possibly preceded by (repeated) applications of the $\neg$ assumption contraction.

- Replace a derivation of the form

$$
\begin{array}{lll}
 & \vdots & \\
i & B & \text{given} \\
i+1 & \boxed{\begin{array}{ll} B & \text{assumption} \\ \vdots & \\ \neg A & \text{given} \end{array}} & \\
j & & \\
j+1 & A \to B & \to I_2\colon\ i,\ i+1\text{–}j \\
j+2 & \neg A & \to E_1\colon\ j+1
\end{array}
$$

with

$$
\begin{array}{lll}
 & \vdots & \\
i & B & \text{given} \\
 & \vdots & \\
i & \neg A & \text{given}
\end{array}
\qquad \text{or} \qquad
\begin{array}{lll}
1. & \boxed{\begin{array}{ll} \bot & \text{assumption} \\ \vdots & \\ B & \text{given} \\ \vdots & \\ \neg A & \text{given} \end{array}} & \\
i & & \\
j & & \\
j+1 & \neg A & \mathit{BOX}\colon\ 1\text{–}j
\end{array}
$$

- Replace a derivation of the form

$$
\begin{array}{lll}
 & \vdots & \\
i & A & \text{given} \\
i+1 & \boxed{\begin{array}{ll} A & \text{assumption} \\ \vdots & \\ \top & \text{given} \end{array}} & \\
j & & \\
j+1 & \neg A & \neg I\colon\ i+1\text{–}j \\
j+2 & \top & \neg E\colon\ i,\ j+1
\end{array}
$$

with

$$
\begin{array}{lll}
 & \vdots & \\
i & A & \text{given} \\
 & \vdots & \\
j & \top & \text{given}
\end{array}
$$

We also have contractions to deal with maximum formulas introduced by CRV.

- Replace a derivation of the form

$$
\begin{array}{lll}
 & \vdots & \\
i & \boxed{\begin{array}{ll} \neg(A \vee B) & \text{assumption} \\ \vdots & \\ \top & \text{given} \end{array}} & \\
j & & \\
j+1 & A \vee B & \mathit{CRV}\colon\ i\text{–}j \\
j+2 & A & \vee E_1\colon\ j+1
\end{array}
$$

with

$$
\begin{array}{rcl}
 & \vdots & \\
i & \neg A & \text{assumption} \\
i+1 & A \vee B & \text{assumption} \\
i+2 & A & \vee E_1\colon\ i+1 \\
i+3 & \top & \neg E\colon\ i,\ i+2 \\
i+4 & \neg(A \vee B) & \neg I\colon\ i+1\text{–}i+3 \\
 & \vdots & \\
j & \top & \text{given} \\
j+1 & A & \mathit{CRV}\colon\ i\text{–}j
\end{array}
$$

The derivation of $\top$ that was previously in the scope of $\neg(A \vee B)$ is now instead in the scope of $\neg A$, but since A is a subformula of $A \vee B$, this does not lead to any issues with newly blocked propositional constants.

- Replace a derivation of the form

$$
\begin{array}{rcl}
 & \vdots & \\
i & \neg(A \wedge B) & \text{assumption} \\
 & \vdots & \\
j & \top & \text{given} \\
j+1 & A \wedge B & \mathit{CRV}\colon\ i\text{–}j \\
j+2 & A & \text{assumption} \\
 & \vdots & \\
k & \top & \text{given} \\
k+1 & B & \text{assumption} \\
 & \vdots & \\
l & \top & \text{given} \\
l+1 & \top & \wedge E\colon\ j+1,\ j+2\text{–}k,\ k+1\text{–}l
\end{array}
$$

with

$$
\begin{array}{rcl}
 & \vdots & \\
i & A \wedge B & \text{assumption} \\
i+1 & A & \text{assumption} \\
 & \vdots & \\
j & \top & \text{given} \\
j+1 & B & \text{assumption} \\
 & \vdots & \\
k & \top & \text{given} \\
k+1 & \top & \vee E\colon\ i,\ i+1\text{–}j,\ j+1\text{–}k \\
k+2 & \neg(A \wedge B) & \neg I\colon\ i\text{–}k+1 \\
 & \vdots & \\
l & \top & \text{given}
\end{array}
\quad .
$$

The subderivation premisses of $\vee E$ are now in the scope of an additional assumption $A \vee B$, which might make necessary applications of the atomic rules no longer legal. We can fix this by moving all those atomic rule applications in the subderivation with assumption A that use propositional constants only occuring in B to before the $A \vee B$ assumption (which must still be in the scope of some assumption to be able to derive $\top$ at all), and vice versa for the subderivation with assumption B. Since two lines moved from different subderivations this way cannot share the same propositional constant, no conflicts occur.

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & \neg(A \to B) & \text{assumption} \\ & \vdots & \\ j & \top & \text{given} \\ j+1 & A \to B & \mathit{CRV}\colon i\text{–}j \\ j+2 & \neg A & \to E_1\colon j+1 \end{array}$$

with

$$\begin{array}{lll} & \vdots & \\ i & A & \text{assumption} \\ i+1 & A \to B & \text{assumption} \\ i+2 & \neg A & \to E_1\colon i+1 \\ i+3 & A & \mathit{Triv}\colon i \\ i+4 & \top & \neg E\colon i+2,\, i+3 \\ i+5 & \neg(A \to B) & \neg I\colon i+1\text{–}i+4 \\ & \vdots & \\ j & \top & \text{given} \\ j+1 & \neg A & \neg I\colon i\text{–}j \end{array}$$

For $\to E_2$, use CRV in place of $\neg I$ to derive B from a subderivation with assumption $\neg B$. The issue of the $\to E_1$ case where the conclusion is sometimes of the same degree as the original maximum formula still remains, and is even more likely to come up now since the result of the contraction always has $\neg A$ derived by $\neg I$, an introduction rule. The workaround remains immediately applying the proper contraction to deal with the new maximum formula in the same step.

- Replace a derivation of the form

$$\begin{array}{lll} & \vdots & \\ i & A & \text{given} \\ i+1 & \neg\neg A & \text{assumption} \\ & \vdots & \\ j & \top & \text{given} \\ j+1 & \neg A & \mathit{CRV}\colon i+1\text{–}j \\ j+2 & \top & \neg E\colon i,\, j+1 \end{array}$$

with

$$\begin{array}{lll}
 & \vdots & \\
i & A & \text{given} \\
i+1 & \neg A & \text{assumption} \\
i+2 & \top & \neg E\colon i,\, i+1 \\
i+3 & \neg\neg A & \neg I\colon i+1\text{–}i+2 \\
 & \vdots & \\
j & \top & \text{given}
\end{array}$$

The rule $\neg E$ is not available outside the scope of all assumptions, so in this case we can be sure of the existence of surrounding assumptions and the final derivation of $\top$ can be safely removed from its subderivation. Even if the derivation continued past line j, we would not need to worry about new operative formulas blocking any inferences afterwards, because having derived $\top$, we can just get any formula by $\top E$ as long as we remain in the same scope.

Replacing the assumption from the original *CRV* application can create a new maximum formula with a higher degree if that assumption was the major premiss of a $\neg E$ application, so we need assumption contractions for such cases.

- Replace a derivation of the form

$$\begin{array}{lll}
 & \vdots & \\
i & \neg(A \vee B) & \text{assumption} \\
 & \vdots & \\
j & A \vee B & \text{given} \\
j+1 & \top & \neg E\colon i,\, j \\
 & \vdots\!\vdots & \\
k & \top & \text{given} \\
k+1 & A \vee B & \mathit{CRV}\colon i,\, k \\
k+2 & A & \vee E_1\colon k+1
\end{array}$$

with

$$\begin{array}{lll}
 & \vdots & \\
i & \neg A & \text{assumption} \\
i+1 & \neg(A \vee B) & \text{assumption} \\
 & \vdots & \\
j & A \vee B & \text{given} \\
j+1 & A & \vee E_1\colon j \\
j+2 & \top & \neg E\colon i,\, j+1 \\
 & \vdots\!\vdots & \\
k & \top & \text{given} \\
k+1 & A \vee B & \mathit{CRV}\colon i+1\text{–}k \\
k+2 & A & \vee E_1\colon k+1 \\
k+3 & \top & \neg E\colon i,\, k+2 \\
k+4 & A & \mathit{CRV}\colon i\text{–}k+3
\end{array}$$

- Replace a derivation of the form

$$
\begin{array}{rl|ll}
 & & \vdots & \\
i & & \neg(A \wedge B) & \text{assumption} \\
 & & \vdots & \\
j & & A \wedge B & \text{given} \\
j+1 & & \top & \neg E\text{: } i,\ j \\
 & & \vdots & \\
k & & \top & \text{given} \\
k+1 & & A \wedge B & \mathit{CRV}\text{: } i\text{–}k \\
k+2 & & A & \text{assumption} \\
 & & \vdots & \\
l & & \top & \text{given} \\
l+1 & & A & \text{assumption} \\
 & & \vdots & \\
m & & \top & \text{given} \\
m+1 & & \top & \wedge E\text{: } k+1,\ k+2\text{–}l,\ l+1\text{–}m
\end{array}
$$

with

$$
\begin{array}{rl|ll}
 & & \vdots & \\
i & & \neg(A \wedge B) & \text{assumption} \\
 & & \vdots & \\
j & & A \wedge B & \text{given} \\
j+1 & & A & \text{assumption} \\
 & & \vdots & \\
k & & \top & \text{given} \\
k+1 & & B & \text{assumption} \\
 & & \vdots & \\
l & & \top & \text{given} \\
l+1 & & \top & \wedge E\text{: } j,\ j+1\text{–}k,\ k+1\text{–}l \\
 & & \vdots & \\
m & & \top & \text{given} \\
m+1 & & A \wedge B & \mathit{CRV}\text{: } i\text{–}m \\
m+2 & & A & \text{assumption} \\
 & & \vdots & \\
n & & \top & \text{given} \\
n+1 & & B & \text{assumption} \\
 & & \vdots & \\
o & & \top & \text{given} \\
o+1 & & \top & \wedge E\text{: } m+1,\ m+2\text{–}n,\ n+1\text{–}o
\end{array}
$$

As we did for the corresponding proper contraction, conflicts between the assumption $\neg(A \vee B)$ and atomic rule applications within the $\wedge E$ subderivations must be circumvented by moving the problematic lines before line i and recovering them with *Triv*.

- Replace a derivation of the form

$$
\begin{array}{rcl}
 & \vdots & \\
i & \neg(A \to B) & \text{assumption} \\
 & \vdots & \\
j & A \to B & \text{given} \\
j+1 & \top & \neg E\text{: } i, j \\
 & \vdots & \\
k & \top & \text{given} \\
k+1 & A \to B & \mathit{CRV}\text{: } i\text{–}k \\
k+2 & \neg A & \to E_1\text{: } k+1
\end{array}
$$

with

$$
\begin{array}{rcl}
 & \vdots & \\
i & A & \text{assumption} \\
i+1 & \neg(A \to B) & \text{assumption} \\
 & \vdots & \\
j & A \to B & \text{given} \\
j+1 & \neg A & \to E_1\text{: } j \\
j+2 & A & \mathit{Triv}\text{: } i \\
j+3 & \top & \neg E\text{: } j+1, j+2 \\
 & \vdots & \\
k & \top & \text{given} \\
k+1 & A \to B & \mathit{CRV}\text{: } i+1\text{–}k \\
k+2 & \neg A & \to E_1\text{: } k+1 \\
k+3 & A & \mathit{Triv}\text{: } i \\
k+4 & \top & \neg E\text{: } k+2, k+3 \\
k+5 & \neg A & \neg I\text{: } i\text{–}k+4
\end{array}
$$

For $\to E_2$, use *CRV* in place of $\neg I$.

- Replace a derivation of the form

$$
\begin{array}{rll}
 & \vdots & \\
i & \neg\neg A & \text{assumption} \\
 & \vdots & \\
j & \neg A & \text{given} \\
j+1 & \top & \neg E\text{: } i, j \\
 & \vdots & \\
k & \top & \text{given} \\
k+1 & \neg A & \mathit{CRV}\text{: } i+1\text{–}j \\
 & \vdots & \\
l & A & \text{given} \\
l+1 & \top & \neg E\text{: } k+1, l
\end{array}
\qquad \text{with} \qquad
\begin{array}{rll}
 & \vdots & \\
i & \neg\neg A & \text{assumption} \\
 & \vdots & \\
j & \neg A & \text{given} \\
 & \vdots & \\
k & A & \text{given} \\
k+1 & \top & \neg E\text{: } j, k \\
 & \vdots & \\
l & \top & \text{given} \\
l+1 & \neg A & \mathit{CRV}\text{: } i\text{–}l \\
 & \vdots & \\
m & A & \text{given} \\
m+1 & \top & \neg E\text{: } l+1, m
\end{array}
$$

Moving the derivation of A into the subderivation is safe here, because $\neg A$ was operative in that portion of the original derivation, and contains the same propositional constants as $\neg\neg A$.

Making *BOX* an introduction rule obviously means we run the risk of having any formula introduced with that rule be a maximum formula if it is the major premiss of some elimination rule. Luckily, the proper contraction to fix this is quite simple and works regardless of the shape of the formula.

- Replace a derivation of the form

$$
\begin{array}{rll}
 & \vdots & \\
i & \bot & \text{assumption} \\
 & \vdots & \\
j & F & \text{given} \\
j+1 & F & \mathit{BOX}\text{: } i\text{–}j \\
 & \vdots & \\
k & G & \text{elimination rule}
\end{array}
$$

with

$$
\begin{array}{rll}
 & \vdots & \\
i & \bot & \text{assumption} \\
 & \vdots & \\
j & F & \text{given} \\
 & \vdots & \\
k & G & \text{elimination rule} \\
k+1 & G & \mathit{BOX}\text{: } i\text{–}k
\end{array}
$$

Note that, after applying the contraction, it is still possible that F is a maximum formula depending on the rule used to introduce it at line j. However, that rule cannot be *BOX* due to the presence of an assumption, so we can resolve the issue with the same trick from above of immediately contracting away the new maximum formula.

There are no contractions that introduce $\wedge E$ applications with conclusions other than $\top$, so just like in the assertional case, we can be sure simplified derivations remain simplified with each contraction step. The same goes for the property of directness, since the only *Triv* applications introduced have as their premiss either assumptions or atomic rule conclusions.

Lemma 6. *If δ is a simplified and direct derivation in $\overline{\mathbf{N}}_{BOX}$ showing $\Gamma \dashv F$, then there exists a simplified and direct $\overline{\mathbf{N}}_{BOX}$ derivation δ' with the same assumptions and conclusion in which no assumption contractions are applicable to maximum formulas of degree $n_{\delta'}$, and $n_{\delta'} = n_\delta$.*

Proof. Because applying an assumption contraction to an appropriate formula in δ (and repeatedly applying it to the additional copy inside the subderivation, if necessary) gives us a derivation with fewer maximum formulas of the same degree that have an applicable assumption contraction, applying these contraction steps to as many maximum formulas of degree n_δ as possible gives us a finite sequence of derivations

$$\delta, \delta^1, \delta^2, \ldots, \delta^k$$

with $n_{\delta^{i+1}} = n_{\delta^i}$, where δ^k has no assumption contractions applicable. Thus, δ^k is the required δ'. □

Lemma 7. *Any simplified and direct derivation in $\overline{\mathbf{N}}_{BOX}$ showing $\Gamma \dashv F$ can be reduced to a normal derivation showing $\Gamma \dashv F$.*

Proof. Let δ_0 be the initial derivation. Then δ_0 contains exactly m_{δ_0} maximum formulas of degree n_{δ_0}, and no maximum formulas with a degree greater than that. By Lemma 6, we can obtain a derivation δ_0' to which no assumption contractions are applicable, which rules out all cases where applying a proper contraction to a maximum formula of degree $n_{\delta_0'}$ $(= n_{\delta_0})$ would produce one of higher degree.

Thus, we can now simply take a maximum formula of degree $n_{\delta_0'}$ in δ_0' and apply the applicable proper contraction to it, resulting in a derivation δ_1 that has either $n_{\delta_1} < n_{\delta_0'}$, or $n_{\delta_1} = n_{\delta_0'}$ and $m_{\delta_1} < m_{\delta_0'}$.

In the first case, we again exhaustively apply the assumptions contractions to get δ_1' and then continue with the next proper contraction to get δ_2. In the second case, we simply proceed to δ_2 in the same way immediately, as there will still be no applicable assumption contractions.

Repeating this process until no maximum formulas are left gives us a finite sequence of derivations $\delta_0, \delta_1, \ldots, \delta_k$ where $\langle n_{\delta_{i+1}}, m_{\delta_{i+1}} \rangle < \langle n_{\delta_i}, m_{\delta_i} \rangle$. For the last element δ_k in

particular, the tuple is $\langle 0, 0 \rangle$, meaning it contains no maximum formulas and is therefore a normal derivation. □

By Lemma 5, a given derivation can be transformed to a simplified and direct derivation, and by Lemma 7, that simplified and direct derivation can be transformed to a normal derivation. This proves Theorem 6.

CHAPTER 5

Adapted Systems for Automation

The assertional and refutational systems of natural deduction we presented so far denote their derivations in a "boxed" style similar to that used by Jaśkowski [17] and Fitch [8]. An essential component of this notation is using the *Triv* rule to repeat previously derived formulas at the positions we need them to apply certain rules, which allows us to save space and improve readability by not needing to re-derive premisses.

However, when constructing a proof in a bottom-up manner, as done in automated systems, that *Triv* rule becomes somewhat problematic: How can we differentiate between a premiss that needs to be derived fully and one that can simply be fetched with *Triv*, if we do not yet know what the lines above the current one will look like once the proof is complete? Certainly one could introduce book-keeping devices that help with making that distinction as information becomes available, but from a programmatic standpoint it is much simpler to follow a Gentzen-style tree notation where there is no relevant ordering of lines and each occurrence of a formula independently includes the full derivation.

Also, while both systems specify the premisses and side conditions required to apply their rules in a sound manner, they do not in any way account for the possibility of loops and infinitely long proofs, which becomes a problem in an automated prover that we ideally want to see terminate under all circumstances. For instance, case distinctions can be nested as many times as we want because the major premiss will still be available in either of the branching subdeductions. To illustrate, consider the attempt at showing $\{p_0 \vee (p_0 \wedge p_1)\} \vdash p_0$ as given in Figure 5.1. Since it is possible to apply $\vee E$ at any level, it is perfectly legal for us to continue nesting as many subdeductions as we want in the omitted portions. Of course, in this simple example, the issue could easily be resolved by just using other rules—*Triv* and $\wedge E$—with higher priority than $\vee E$, but that would not prevent an automated prover from running into endless derivations of this form while pursuing failing branches. This in turn would mean to never backtrack to find the correct path, so we need to put measures in place to more strictly disallow pointless repetitions.

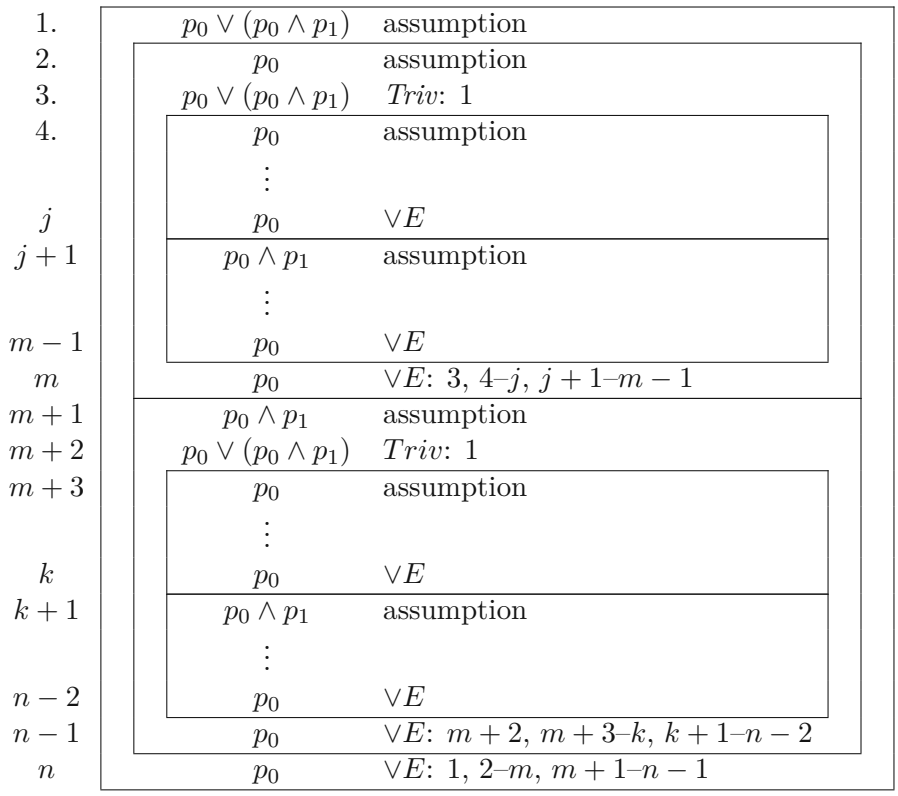


Figure 5.1: A derivation for showing $\{p_0 \vee (p_0 \wedge p_1)\} \vdash p_0$.

5.1 Preliminaries

Before introducing the adapted calculi that address the concerns stated above, we will discuss the general structure that will be used for both the assertional and refutational case.

As seen in already existing natural deduction provers from the literature we reviewed in Chapter 3, the general approach is to apply some rules bottom-up from the conclusion and others top-down from the assumptions, constructing two or more partial derivation trees that can then be joined together at common formulas. The design of our calculi will also follow this pattern, but with the notable special property that the premisses of top-down rules never require bottom-up rules in their derivations, and the crossover between the two sets of rules thus goes in only one direction. This makes the systems trivially loop-free so long as both the top-down and bottom-up rules are each loop-free in isolation.

In what follows, we will use a Gentzen-style tree notation, but also follow Ferrari and

Fiorentini [5] in using labeled sequents rather than formulas for the nodes. These sequents take the basic form $\Gamma \vdash F\circ$ for the assertional system and $\Gamma \dashv F\circ$ for the refutational one, with Γ denoting the currently active assumptions, F the formula derived at that node, and $\circ \in \{\uparrow, \downarrow\}$ labeling the sequent as being part of either the bottom-up ($\uparrow$) or top-down ($\downarrow$) derivation. In the actual systems, the sequents also include components for additional information that needs to be tracked throughout the derivation, but since the requirements differ between the assertional and refutational case, we will omit these components for now.

For example, using the basic notation just introduced, the $\wedge I$ and $\rightarrow I$ rules of **N** would be written as follows:

$$\frac{\Gamma \vdash A\uparrow \quad \Gamma \vdash B\uparrow}{\Gamma \vdash A \wedge B\uparrow}\ \wedge I \qquad\qquad \frac{A, \Gamma \vdash B\uparrow}{\Gamma \vdash A \rightarrow B\uparrow}\ \rightarrow I$$

Similarly, $\wedge E_1$ and $\rightarrow E$ become:

$$\frac{\Gamma \vdash A \wedge B\downarrow}{\Gamma \vdash A\downarrow}\ \wedge E_1 \qquad\qquad \frac{\Gamma \vdash A \rightarrow B\downarrow \quad \Gamma \vdash A\downarrow}{\Gamma \vdash B\downarrow}\ \rightarrow E$$

In the $\rightarrow I$ example, we can see a simplified notation $A, \Gamma \vdash B$ instead of $\{A\} \cup \Gamma \vdash B$. If $\Gamma = \emptyset$, this sequent could also be written $A \vdash B$. To save some space and improve readability, we will consistently use this notation for sets that occur in sequents.

5.2 The Assertional Case

The loop-free counterpart of the assertional natural deduction calculus **N**, denoted by $\mathbf{N}^*$, consists of the following inference rules.

$$\frac{F \in \Gamma}{\Gamma \vdash F\downarrow}\ Asm \qquad\qquad \frac{\Gamma \vdash F\downarrow}{\Gamma \vdash F; \Delta\uparrow}\ \downarrow\uparrow$$

The simplest rules are those that allow us to use assumptions in top-down reasoning and top-down conclusions in bottom-up reasoning, thereby providing the joining points between the multiple sources of information.

The extra component Δ on the bottom-up side of $\downarrow\uparrow$ exists for loop avoidance and contains the names of rules we want "blocked" in that derivation. This is most relevant when using the $\neg E$ rule in a bottom-up derivation:

$$\frac{\Gamma \vdash \neg F; \Delta\downarrow \quad \Gamma \vdash F; \langle\neg E\rangle, \Delta\uparrow \quad \langle\neg E\rangle \notin \Delta}{\Gamma \vdash \bot; \Delta\uparrow}\ \neg E$$

Without Δ, certain corner cases that allow an exact copy of the conclusion sequent to appear in the bottom-up reasoning of the minor premiss could easily lead to an infinite

loop, as $\neg E$ would be applicable again with the same premisses. We can in fact go as far as disabling the rule entirely to avoid this, since ultimately there is no need to ever derive $\bot$ twice under the same assumptions.

$$\frac{}{\Gamma \vdash \top; \Delta \uparrow} \top I \qquad \frac{\Gamma \vdash A; \Delta \uparrow \quad \Gamma \vdash B; \Delta \uparrow}{\Gamma \vdash A \wedge B; \Delta \uparrow} \wedge I$$

$$\frac{\Gamma \vdash A; \Delta \uparrow}{\Gamma \vdash A \vee B; \Delta \uparrow} \vee I_1 \qquad \frac{\Gamma \vdash B; \Delta \uparrow}{\Gamma \vdash A \vee B; \Delta \uparrow} \vee I_2$$

$$\frac{A, \Gamma \vdash B; \emptyset \uparrow \quad A \notin \Gamma}{\Gamma \vdash A \to B; \Delta \uparrow} \to I \qquad \frac{\Gamma \vdash B; \Delta \uparrow \quad A \in \Gamma}{\Gamma \vdash A \to B; \Delta \uparrow} \to I^*$$

The above introduction rules function just like their counterparts in **N**, with no particular side conditions to consider. However, $\to I$ has some noteworthy special features: it is split into two variants, $\to I$ and $\to I^*$, to avoid the introduction of redundant assumptions, and when it does introduce an assumption, Δ is reset.

The rules that model indirect proofs do require some additional side conditions to cut off useless derivations:

$$\frac{\neg F, \Gamma \vdash \bot; \emptyset \uparrow \quad F \neq \bot \quad F \neq \neg F' \quad \neg F \notin \Gamma}{\Gamma \vdash F; \Delta \uparrow} CRF \qquad \frac{F, \Gamma \vdash \bot; \emptyset \uparrow \quad F \notin \Gamma}{\Gamma \vdash \neg F; \Delta \uparrow} \neg I$$

For the elimination rules, only the $\wedge$ ones can be straightforwardly adapted into top-down rules.

$$\frac{\Gamma \vdash A \wedge B \downarrow}{\Gamma \vdash A \downarrow} \wedge E_1 \qquad \frac{\Gamma \vdash A \wedge B \downarrow}{\Gamma \vdash B \downarrow} \wedge E_2$$

On the other hand, $\vee E$ requires two subdeductions as its minor premisses, making it rather unsuitable for top-down reasoning. Therefore, despite being an elimination rule, we adapt it into a bottom-up rule.

$$\frac{\Gamma \vdash A \vee B; \Delta \downarrow \quad A, \Gamma \vdash F; \emptyset \uparrow \quad B, \Gamma \vdash F; \emptyset \uparrow \quad A \notin \Gamma \quad B \notin \Gamma}{\Gamma \vdash F; \Delta \uparrow} \vee E$$

In view of the normalisation theorem, we can be sure the major premiss will only require top-down reasoning, but the minor premisses still need to be processed bottom-up.

Ordinarily, the same would go for $\to E$ and its minor premiss, but we can actually use it as a pure top-down rule by supplementing it with a second $\to$ elimination rule:

$$\frac{\Gamma \vdash A \to B \downarrow \quad \Gamma \vdash A \downarrow}{\Gamma \vdash B \downarrow} \to E \qquad \frac{\Gamma \vdash A \to B \downarrow}{\Gamma \vdash \neg A \vee B \downarrow} \to E_\vee$$

Cases where $\to E$ would require bottom-up rules to derive its minor premiss can also be handled by a combination of $\to E_\vee$ and $\vee E$. While this roundabout method diminishes the natural character of the system somewhat, it is very useful for avoiding loops, as it removes the ability to cross from top-down reasoning into bottom-up reasoning.

Finally, $\bot E$ is adapted as a bottom-up rule simply because it can derive an infinite amount of formulas (all of them, in fact) once the premiss $\bot$ is present, which is not particularly conducive to efficient top-down reasoning.

$$\frac{\Gamma \vdash \bot; \langle \bot E \rangle, \Delta \uparrow \quad F \neq \bot \quad \langle \bot E \rangle \notin \Delta}{\Gamma \vdash F; \Delta \uparrow} \bot E$$

The side conditions serve to explicitly disallow pointless repetition such as deriving $\bot$ from $\bot$, or using a $\bot E$ conclusion just to reach another $\bot$.

5.2.1 Soundness

As soundness and completeness of **N** are known, the same properties can be proven for $\mathbf{N}^*$ via a translation to and from the original system. In the case of soundness, we must show the "to" direction.

Theorem 7. *For any derivation in* $\mathbf{N}^*$ *with root sequent* $\Gamma \vdash F; \Delta \uparrow$*, there exists a derivation in* **N** *whose last line contains the formula* F *with active assumptions* Γ*.*

Proof. The translation is straightforward, since most rules of $\mathbf{N}^*$ also exist in **N** with the same premisses and no side conditions. We can omit *Asm* and $\downarrow\uparrow$ entirely since **N** treats all formulas the same, regardless of how they were derived (although in some cases, *Asm* may need to be translated to a *Triv* rule to fetch the assumption from the start of the subdeduction). For applications of $\to I^*$, we can easily wrap the derivation of the premiss in a subdeduction to enable $\to I$, since an additional assumption—let alone an additional copy of one that is already present—cannot interfere with any rules of **N**. Finally, the additional rule $\to E_\vee$ can be replaced with this derivation:

$$\begin{array}{cc|l}
 & \vdots & \\
i & A \to B & \text{given} \\
\hline
i+1 & \neg(\neg A \vee B) & \text{assumption} \\
\hline
i+2 & \neg A & \text{assumption} \\
i+3 & \neg A \vee B & \vee I_1\colon\ i+2 \\
i+4 & \bot & \neg E\colon\ i+1,\ i+3 \\
\hline
i+5 & A & \mathit{CRF}\colon\ i+2\text{–}i+4 \\
i+6 & B & \to E\colon\ i,\ i+5 \\
i+7 & \neg A \vee B & \vee I_2\colon\ i+6 \\
i+8 & \bot & \neg E\colon\ i+1,\ i+7 \\
\hline
i+9 & \neg A \vee B & \mathit{CRF}\colon\ i+1\text{–}i+8
\end{array}$$

□

5.2.2 Completeness

To show completeness requires proving the "from" direction of the translation, which is the more difficult one since we need to account for derivations that do not obey the added side conditions. To simplify the task, we first ensure some flexibility with regards to Γ and Δ.

Lemma 8. *Given a derivation in* $\mathbf{N}^*$ *with assumptions* Γ *and formula* F *at the root, and a set of formulas* $\Gamma' \supseteq \Gamma$*, there exists a derivation in* $\mathbf{N}^*$ *with assumptions* Γ' *and formula* F *at the root.*

Proof. The proof proceeds by induction on the depth d of the given derivation, defined as the number of rule applications in the longest branch.

Base Case: $d = 1$. Thus, the root rule is either $\top I$ or *Asm*. The former obviously works with any assumptions since it has neither premisses nor side conditions, and the latter only requires $F \in \Gamma'$, which follows from the known $F \in \Gamma$.

Induction Hypothesis: For any derivation in $\mathbf{N}^*$ with depth $1 < d \leq n - 1$ that has assumptions Γ and formula F at the root, there exists a derivation in $\mathbf{N}^*$ with assumptions Γ' and formula F at the root, where $\Gamma' \supset \Gamma$.

Induction Step: We distinguish by the root rule.

- $\to I$: If $A \in \Gamma'$, simply replace the root rule with $\to I^*$, which becomes usable in just such a case.
- *CRF*, $\neg I$: If Γ' does not contain $\neg F$ or F, respectively, we can use it as substitute for Γ without violating the side condition and obtain a sound derivation. Otherwise, we have in the case of *CRF* that $\neg F, \Gamma' \vdash \bot; \emptyset \uparrow$ by the induction hypothesis, which by contraction gives us $\Gamma' \vdash \bot; \emptyset \uparrow$ since $\neg F \in \Gamma'$. This premiss can be used to derive $\Gamma' \vdash F; \emptyset \uparrow$ by $\bot E$, and the same approach works for $\neg I$. If $\bot E$ cannot be used because there is already an application of it in the derivation of $\Gamma' \vdash \bot; \emptyset \uparrow$, it suffices to replace the conclusion of that application by F.
- $\vee E$: Suppose $A \in \Gamma'$. By the induction hypothesis, we have a derivation for $A, \Gamma' \vdash F; \emptyset \uparrow$ and hence $\Gamma' \vdash F; \emptyset \uparrow$, which is exactly what we need. If $B \in \Gamma'$, the same reasoning can be applied, and if $A, B \notin \Gamma'$, we can simply replace all occurrences of Γ with Γ' and not violate any side conditions.
- The remaining rules do not have side conditions that could conflict with additional assumptions, so we can always just replace Γ with Γ' and be done.

□

Lemma 9. *Given a derivation in* $\mathbf{N}^*$ *with root sequent* $\Gamma \vdash F; \Delta \uparrow$ *and a set of rule names* $\Delta' \subseteq \Delta$ *, there exists a derivation in* $\mathbf{N}^*$ *with root sequent* $\Gamma \vdash F; \Delta' \uparrow$.

Proof. Since $\neg E$ and $\bot E$ are the only rules that interact with Δ, and only to check for the absence of a rule name, replacing all occurences Δ in the given derivation with Δ' produces a sound derivation with the desired conclusion. □

By the normalisation theorem, there is a corresponding normal derivation for any derivation in **N**, so it is sufficient to only consider those normal derivations for our proof of completeness.

Theorem 8. *For any (simplified and direct) normal derivation in* **N** *whose last line contains the formula F with active assumptions Γ, there exists a derivation in* $\mathbf{N}^*$ *with root sequent $\Gamma \vdash F; \emptyset \uparrow$.*

Proof. We proceed by induction on the length l of the **N** derivation.

Base Case: $l = 1$. Then, the formula F is either an assumption (and thus $F \in \Gamma$) or was derived by $\top I$. We get this $\mathbf{N}^*$ derivation for the first case:

$$\dfrac{\dfrac{F \in \Gamma}{\Gamma \vdash F \downarrow}\, Asm}{\Gamma \vdash F; \emptyset \uparrow}\, \downarrow\uparrow$$

And this one for the second:

$$\dfrac{}{\Gamma \vdash \top; \emptyset \uparrow}\, \top I$$

Induction Hypothesis: For any normal derivation in **N** with length $1 < l \leq n-1$ that has formula F and active assumptions Γ at line l, there exists a derivation in $\mathbf{N}^*$ with root sequent $\Gamma \vdash F; \emptyset \uparrow$.

We further assert that any derivation in **N** that ends on an elimination rule corresponds to a derivation in $\mathbf{N}^*$ with root rule $\downarrow\uparrow$, $\vee E$, or $\bot E$.

Induction Step: We distinguish based on the rule used to derive the formula at line n, which must be a rule with one or more premisses.

- *Triv*: The premiss is exactly the formula F in the scope of the assumptions in Γ or some subset thereof, so by the induction hypothesis and Γ-weakening, we automatically have a derivation in $\mathbf{N}^*$ with root sequent $\Gamma \vdash F; \emptyset \uparrow$.
- $\wedge I_1$, $\wedge I_2$, $\vee I_1$, $\vee I_2$, $\top I$: For all of these rules, $\mathbf{N}^*$ provides rules that require the exact same premisses as bottom-up sequents with no side conditions, so extending the derivation from those we have by the induction hypothesis is trivial.
- $\to I$: If $A \notin \Gamma$, we have the required premiss for $\to I$ by the induction hypothesis. If instead $A \in \Gamma$, we can use $\to I^*$, which has its premiss available by contraction.

- $\bot E$: We have by the induction hypothesis a derivation ending on the sequent $\Gamma \vdash \bot; \emptyset \uparrow$. If $F = \bot$, this already is the derivation we want, but otherwise we need to change its root sequent to $\Gamma \vdash \bot; \langle \bot E \rangle \uparrow$, which is no problem if the derivation contains no application of $\bot E$ under the assumptions Γ. If it does, however, that step must itself have the premiss $\Gamma \vdash \bot; \langle \bot E \rangle \uparrow$ (or one that reduces to it by Δ-weakening), and we merely need to swap out its conclusion.

- $\wedge E_1, \wedge E_2$: In either case, we have the formula $A \wedge B$ as the sole premiss, which in a normal derivation must have been obtained with an elimination rule. Thus the $\mathbf{N}^*$ derivation we get from the induction hypothesis has as its root rule either $\downarrow\uparrow$, $\bot E$, or $\vee E$. In the first case we have the sequent $\Gamma \vdash A \wedge B \downarrow$ to easily extend the derivation and in the second we can just swap out the conclusion of $\bot E$. The final case is significantly more complex and jointly involves multiple elimination rules, so we will disregard it for now.

- $\to E$: The major premiss is $A \to F$ and the minor premiss A. Since we are working with a normal derivation, $A \to F$ must have been derived by an elimination rule, so the induction hypothesis provides us with a $\mathbf{N}^*$ derivation with root sequent $\Gamma \vdash A \to F; \emptyset \uparrow$ and root rule $\downarrow\uparrow$, $\vee E$, or $\bot E$. Also by the induction hypothesis, we have a $\mathbf{N}^*$ derivation of $\Gamma \vdash A; \emptyset \uparrow$, but its root rule can be anything.

  There are several cases to consider here, depending on the $\mathbf{N}^*$ rule by which the premisses are derived. In the simplest case, either of the two is $\bot E$, as that means the derivation also includes $\Gamma \vdash \bot; \langle \bot E \rangle \uparrow$ and we can build a derivation for F like this:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash \bot; \langle \bot E \rangle \uparrow}\,\text{given}}{\Gamma \vdash F; \emptyset \uparrow}\,\bot E$$

  Otherwise, we are looking at one of the following combinations.

  - Both derived by $\downarrow\uparrow$. Then the derivations must include the sequents $\Gamma \vdash A \to F \downarrow$ and $\Gamma \vdash A \downarrow$, enabling this derivation of F.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash A \to F \downarrow}\,\text{given} \quad \dfrac{\vdots}{\Gamma \vdash A \downarrow}\,\text{given}}{\Gamma \vdash F \downarrow}\,\to E}{\Gamma \vdash F; \emptyset \uparrow}\,\downarrow\uparrow$$

  - $A \to F$ derived by $\downarrow\uparrow$, A derived by a different rule. In this case, we can get to F via $\vee E$. First, let δ_A be the derivation

$$\dfrac{\dfrac{\dfrac{\neg A \in \neg A, \Gamma}{\neg A, \Gamma \vdash \neg A \downarrow}\,Asm \quad \dfrac{\vdots}{\neg A, \Gamma \vdash A; \langle \neg E \rangle, \langle \bot E \rangle \uparrow}\,\text{given}}{\neg A, \Gamma \vdash \bot; \langle \bot E \rangle \uparrow}\,\neg E}{\neg A, \Gamma \vdash F; \emptyset \uparrow}\,\bot E \quad .$$

Then we have the following derivation that ends on $\Gamma \vdash F; \emptyset \downarrow$:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash A \to F \downarrow}\,\text{given}}{\Gamma \vdash \neg A \vee F \downarrow}\,{\to}E_{\vee} \quad \delta_A \quad \dfrac{\dfrac{F \in F, \Gamma}{F, \Gamma \vdash F \downarrow}\,Asm}{F, \Gamma \vdash F; \emptyset \uparrow}\,{\downarrow}{\uparrow}}{\Gamma \vdash F; \emptyset \uparrow}\,{\vee}E$$

While we only get $\Gamma \vdash A; \emptyset \uparrow$ via the induction hypothesis, δ_A uses $\neg A, \Gamma \vdash A; \langle \neg E \rangle, \langle \bot E \rangle \uparrow$. The additional assumption is not a problem since the system permits Γ-weakening, but being unable to use $\neg E$ and $\bot E$ under the same assumptions may conflict with the contents of the given derivation. However, if that derivation were to contain such an application of either rule, it would already have $\neg A, \Gamma \vdash \bot; \Delta \uparrow$ as its conclusion or premiss ($\Delta = \emptyset$ or $\Delta = \langle \bot E \rangle$), thus eliminating the need for a derivation of A entirely.

Another potential issue are the side conditions on ${\vee}E$, which would be violated if $\neg A \in \Gamma$ and/or $F \in \Gamma$. In such a case, the derivation from any subdeduction whose assumption is already present can simply be used to directly obtain F.

– $A \to F$ derived by ${\vee}E$, A derived by any rule. This case will be addressed later.

- ${\vee}E$: Since the normal derivations we are working with are also simplified, we know that $F = \bot$ here. The induction hypothesis gives us a $\mathbf{N}^*$ derivation for $\Gamma \vdash A \vee B; \emptyset \uparrow$ ending on ${\downarrow}{\uparrow}$, ${\vee}E$, or $\bot E$, as well as for $A, \Gamma \vdash \bot; \emptyset \uparrow$ and $B, \Gamma \vdash \bot; \emptyset \uparrow$ ending on any rule. The case where the major premiss was derived by $\bot E$ is especially trivial here, and in the ${\downarrow}{\uparrow}$ case we can also simply extend the derivation, as seen below.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash A \vee B \downarrow}\,\text{given} \quad \dfrac{\vdots}{A, \Gamma \vdash \bot; \emptyset \uparrow}\,\text{given} \quad \dfrac{\vdots}{B, \Gamma \vdash \bot; \emptyset \uparrow}\,\text{given}}{\Gamma \vdash \bot; \emptyset \uparrow}\,{\vee}E$$

If $A \in \Gamma$ or $B \in \Gamma$, we cannot apply the ${\vee}E$ rule of $\mathbf{N}^*$, but we also do not need to: By the induction hypothesis, we have $A, \Gamma \vdash \bot; \emptyset \uparrow$ and if $A \in \Gamma$ that means by contraction $\Gamma \vdash \bot; \emptyset \uparrow$, which is what we ultimately want to derive.

- $\neg E$: By the induction hypothesis, we have the sequents $\Gamma \vdash F; \emptyset \uparrow$ and $\Gamma \vdash \neg F; \emptyset \uparrow$, the latter derived by ${\downarrow}{\uparrow}$, ${\vee}E$, or $\bot E$. If ${\uparrow}{\downarrow}$, the next inference step is straightforward:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash \neg F \downarrow}\,\text{given} \quad \dfrac{\vdots}{\Gamma \vdash F; \langle \neg E \rangle \uparrow}\,\text{given}}{\Gamma \vdash \bot; \emptyset \uparrow}\,\neg E$$

However, the addition of $\langle \neg E \rangle$ in the Δ component of the minor premiss may conflict with the derivation of F we are given. In such a case, there is another $\neg E$

application further up concluding $\Gamma \vdash \bot; \emptyset \uparrow$, which is exactly the conclusion we want to reach anyway. With $\Gamma \vdash \neg F; \emptyset \uparrow$ derived by $\bot E$, we again simply need to use the premiss of that rule directly, and we will investigate the $\vee E$ case later.

- $\neg I$: If the side condition $F \notin \Gamma$ holds, we can simply apply $\neg I$ to the available premiss $F, \Gamma \vdash \bot; \emptyset \uparrow$. Otherwise, that same premiss gives us $\Gamma \vdash \bot; \emptyset \uparrow$ by contraction, enabling the $\bot E$ rule.

- *CRF*: The cases where the side conditions hold and where only $\neg F \notin \Gamma$ is violated work just like they did with $\neg I$. At first glance, the remaining conditions $F \neq \bot$ and $F \neq \neg F'$ seem harmless—the assumption $\neg\bot$ would not be useful anyway, and $\neg I$ already handles deriving negations by contradiction. However, we should consider some corner cases where an assumption $\neg\bot$ or $\neg\neg F'$ could potentially interact with other formulas in useful ways.

  - $F = \bot$, $\Gamma = \{\neg\bot \to G, \neg G\}$. If we could use *CRF* with assumption $\neg\bot$, we could unlock G for $\neg E$ using $\to E$, precisely because we have that specific assumption. There is, however, also a derivation that does not use *CRF* at all. Let δ be

$$\dfrac{\dfrac{\neg\neg\bot \in \neg\neg\bot, \Gamma}{\neg\neg\bot, \Gamma \vdash \neg\neg\bot \downarrow}\, Asm \qquad \dfrac{\dfrac{\bot \in \bot, \neg\neg\bot, \Gamma}{\bot, \neg\neg\bot, \Gamma \vdash \bot \downarrow}\, Asm}{\bot, \neg\neg\bot, \Gamma \vdash \bot; \emptyset \uparrow}\, \downarrow\uparrow}{\dfrac{}{\neg\neg\bot, \Gamma \vdash \neg\neg\bot; \langle\neg E\rangle \uparrow}\, \neg I}\, \neg E.$$

    We can then construct the following derivation.

$$\dfrac{\dfrac{\dfrac{\neg\bot \to G \in \Gamma}{\Gamma \vdash \neg\bot \to G; \emptyset \downarrow}\, Asm}{\Gamma \vdash \neg\neg\bot \vee G \downarrow}\, \to E_\vee \qquad \delta \qquad \dfrac{\neg G \in G, \Gamma}{G, \Gamma \vdash \neg G; \emptyset \uparrow}\, Asm \qquad \dfrac{\dfrac{G \in G, \Gamma}{G, \Gamma \vdash G \downarrow}\, Asm}{G, \Gamma \vdash G; \langle\neg E\rangle \uparrow}\, \downarrow\uparrow}{\Gamma \vdash \bot; \emptyset \uparrow}\, \vee E \qquad \neg E$$

  - $F = \neg F'$, $\Gamma = \neg\neg F' \to \bot$. Again, the required $\bot$ is only unlockable using the specific assumption *CRF* would produce if used disregarding its side condition, and $\neg I$ seemingly fails. But we can construct a roundabout alternate derivation, beginning from a derivation δ':

$$\dfrac{\dfrac{\dfrac{\neg F' \in \neg F', \neg\neg\neg F', F', \Gamma}{\neg F', \neg\neg\neg F', F', \Gamma \vdash \neg F' \downarrow}\, Asm \qquad \dfrac{\dfrac{F' \in \neg F', \neg\neg\neg F', F', \Gamma}{\neg F', \neg\neg\neg F', F', \Gamma \vdash F' \downarrow}\, Asm}{\neg F', \neg\neg\neg F', F', \Gamma \vdash F'; \langle\neg E\rangle \uparrow}\, \downarrow\uparrow}{\neg F', \neg\neg\neg F', F', \Gamma \vdash \bot; \emptyset \uparrow}\, \neg E}{\neg\neg\neg F', F', \Gamma \vdash \neg\neg F'; \langle\neg E\rangle \uparrow}\, \neg I$$

Using δ', we build the derivation δ as

$$\dfrac{\dfrac{\neg\neg\neg F' \in \neg\neg\neg F', F', \Gamma}{\neg\neg\neg F', F', \Gamma \vdash \neg\neg\neg F' \downarrow}\ Asm \qquad \delta'}{\neg\neg\neg F', F', \Gamma \vdash \bot; \emptyset \uparrow}\ \neg E,$$

and finally we use δ in the full derivation of $\Gamma \vdash \neg F'; \emptyset \uparrow$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg\neg F' \to \bot \in F', \Gamma}{F', \Gamma \vdash \neg\neg F' \to \bot; \emptyset \downarrow}\ Asm}{F', \Gamma \vdash \neg\neg\neg F' \vee \bot \downarrow}\ \to E_\vee \qquad \delta \qquad \dfrac{\dfrac{\bot \in \bot, F', \Gamma}{\bot, F', \Gamma \vdash \bot \downarrow}\ Asm}{\bot, F'.\Gamma \vdash \bot; \emptyset \uparrow}\ \downarrow\uparrow}{F', \Gamma \vdash \bot; \emptyset \uparrow}\ \vee E}{\Gamma \vdash \neg F'; \emptyset \uparrow}\ \neg I$$

We still have to handle the cases where the major premiss of an elimination rule (specifically, $\wedge E_1$, $\wedge E_2$, $\to E$, $\vee E$, or $\neg E$) is only made available by the induction hypothesis as a bottom-up sequent derived by $\vee E$. Since any applications of $\vee E$ and $\neg E$ in the original **N** derivation can only have conclusion $\bot$, which cannot have any further elimination rules applied to it, it follows that a major premiss $G_{k+1} \neq \bot$ derived by $\vee E$ after translation must originally have come from an application of $\wedge E_1$, $\wedge E_2$, or $\to E$, in turn requiring a major premiss of the form $G_{k+1} \wedge G_k$, $G_k \wedge G_{k+1}$, or $G_k \to G_{k+1}$.

In the simplest case, we have a top-down sequent for this formula, however, we must also consider the possibility that its derivation was already transformed to end on $\vee E$. We established above that the two $\wedge E$ rules do not require such a transformation if they have a top-down sequent for their major premiss, so another transformation to $\vee E$ must have happened further up in the derivation to make it necessary. On the other hand, a transformation of $\to E$ can also be caused by not having a top-down sequent for the minor premiss. Following this train of thought to its logical conclusion, we must by the induction hypothesis have the sequents $\Gamma \vdash G_0 \to G^1 \downarrow$ (with $G^i = (G_i \circ_i \cdots \circ_k G_{k+1})$, $\circ_i \in \{\wedge, \to\}$, for $i \in \{1, \ldots, k\}$) and $\Gamma \vdash G_0; \emptyset \uparrow$, which are the premisses of an initial $\to E$ application transformed to $\vee E$. To be precise, G^1 can have other forms depending on which $\wedge E$ rules are applied to decompose the formula as the derivation proceeds, but for ease of notation, we will assume it is only $\wedge E_2$ so the subformula of interest is always on the right side. We also have access to sequents $\Gamma \vdash G_i; \emptyset \uparrow$ for every i where $\circ_i = \to$, since these connectives must originally have been eliminated via $\to E$. G_{k+1} itself, being the major premiss of the elimination rule we need to transform, is one of $F \wedge B$, $A \wedge F$, $A \to F$, $A \vee B$, or $\neg A$ ($F = \bot$ in the last two cases).

Now recall the way $\to E$ is transformed to $\vee E$ if the minor premiss is only available as a bottom-up sequent. The derivation δ_A is

$$\dfrac{\dfrac{\dfrac{\neg A \in \neg A, \Gamma}{\neg A, \Gamma \vdash \neg A \downarrow}\ Asm \qquad \dfrac{\vdots}{\neg A, \Gamma \vdash A; \langle \neg E \rangle, \langle \bot E \rangle \uparrow}\ \text{given}}{\neg A, \Gamma \vdash \bot; \langle \bot E \rangle \uparrow}\ \neg E}{\neg A, \Gamma \vdash F; \emptyset \uparrow}\ \bot E$$

and used as part of the full transformed derivation:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash A \to F\downarrow}\,\text{given}}{\Gamma \vdash \neg A \vee F\downarrow}\to E_\vee \quad \delta_A \quad \dfrac{\dfrac{F \in F, \Gamma}{F, \Gamma \vdash F\downarrow}\,Asm}{F, \Gamma \vdash F; \emptyset\uparrow}\downarrow\uparrow}{\Gamma \vdash F; \emptyset\uparrow}\vee E,$$

We observe that such a derivation can easily have its conclusion F swapped out for F', provided F' is the conclusion of an applicable elimination rule with F as its major premiss. Obviously δ_A can have its conclusion changed arbitrarily, and in the other subdeduction, F is available as an assumption and thus can be used for any rule. This method of replacement works even if we exchange the right subdeduction with another nested derivation of the same form (because all we need to do is change its conclusion to F' by this very method), and continues working to any nesting depth.

Equipped with all this information, we can produce a $\mathbf{N}^*$ derivation of our desired conclusion F. The root rule is $\vee E$ and the structure is fairly similar to the transformed $\to E$ application shown above, with the first minor premiss being a familiar-looking derivation δ_1:

$$\dfrac{\dfrac{\dfrac{G_0 \in \neg G_0, \Gamma}{\neg G_0, \Gamma \vdash \neg G_0\downarrow}\,Asm \quad \dfrac{\vdots}{\neg G_0, \Gamma \vdash G_0; \langle \neg E\rangle, \langle \bot E\rangle\uparrow}\,\text{given}}{\neg G_0, \Gamma \vdash \bot; \langle \bot E\rangle\uparrow}\neg E}{\neg G_0, \Gamma \vdash F; \emptyset\uparrow}\bot E.$$

However, the derivation for the second minor premiss, δ_2, takes slightly different forms depending on the elimination rule originally used to derive F and how many transformations to $\vee E$ were required along the way. In any case, it consists of some sequence of $\wedge E_1$, $\wedge E_2$, $\to E$, and/or $\neg E$ used to derive F from G^j ($j \in \{1, \ldots, k\}$), nested within the right subdeduction of zero or more $\vee E$ applications structured like one obtained from transforming $\to E$. This derivation can be obtained by deriving G^1 from $G_0 \to G^1$ as we did in the basic $\to E$ case and then repeatedly swapping out the $\vee E$ conclusion G^i for G^{i+1} (which follows from G^i by $\to E$ or $\wedge E_2$), finally going from G^k to just G_{k+1}, and from there to F. Put together, the full derivation is

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash G_0 \to G\downarrow}\,\text{given}}{\Gamma \vdash \neg G_0 \vee G\downarrow}\to E_\vee \quad \delta_1 \quad \delta_2}{\Gamma \vdash F; \emptyset\uparrow}\vee E.$$

In this derivation as well, we could replace F with any F' that follows from it by an elimination rule, as the root rule is $\vee E$, the root rule of δ_1 is $\bot E$, and δ_2 is a derivation of the same form. □

5.2.3 Loop-freeness

When following a derivation in $\mathbf{N}^*$ from the root towards the leaves, any branch in which a top-down rule occurs cannot have any bottom-up rules further up, as there is no way to cross from top-down reasoning back into bottom-up reasoning. We can therefore be assured our adapted system does not allow for infinite loops as long as none can occur with only the bottom-up rules or only the top-down rules.

We first consider the bottom-up rules. $\top I$ and $\downarrow\uparrow$ obviously leave no room for loops simply by not having any premisses that can be derived by bottom-up rules, while $\wedge I$, $\vee I_1$, $\vee I_2$, $\to I$, and $\to I^*$ are limited in their repetition because the formulas in their premisses are subformulas of those in the conclusion.

However, this is not the case for $\neg E$, which can reintroduce an arbitrary amount of complexity in its minor formula F. One could therefore imagine a scenario in which a sequent with $\Gamma \vdash \bot$ occurs again in this branch, allowing application of $\neg E$ with the same premisses and thus an infinite loop:

$$\dfrac{\dfrac{\neg(p \vee \bot) \in \{\neg(p \vee \bot), p\}}{\{\neg(p \vee \bot), p\} \vdash \neg(p \vee \bot) \downarrow}\ \mathit{Asm} \quad \dfrac{\dfrac{\vdots}{\{\neg(p \vee \bot), p\} \vdash \bot; \langle \neg E \rangle \uparrow}\ \neg E?}{\{\neg(p \vee \bot), p\} \vdash p \vee \bot; \langle \neg E \rangle \uparrow}\ \vee I_2}{\{\neg(p \vee \bot), p\} \vdash \bot; \emptyset \uparrow}\ \neg E$$

But as we can see, the $\langle \neg E \rangle$ added to the Δ component of the sequents in the minor premiss successfully handles this case by blocking the repeated application of $\neg E$. Instead of getting lost on a failing search path, an automated prover would now be forced to backtrack and try another rule such as $\vee I_1$ instead, which quickly completes the derivation.

For CRF, $\neg I$, and $\vee E$, the side conditions preventing repeated assumptions serve to prevent loops. Each of these rules can only be reapplied within a subdeduction among its premisses if doing so would actually introduce a fresh assumption. This means CRF and $\neg I$ cannot be reused for the same formula, and $\vee E$ cannot be applied with the same major formula $A \vee B$. The restriction to fresh assumptions is also important because every new assumption resets Δ to $\emptyset$, so if there was a way to introduce them infinitely, it would easily be possible to construct, for instance, a loop of repeated $\bot E$ applications in deeper and deeper subdeductions. This is the reason why $\to I$, despite already reducing complexity bottom-up, must also be unable to repeat assumptions and instead let $\to I^*$ handle such cases.

The loop-free nature of the top-down rules is apparent if we take into account that they are applied starting from the assumptions, and their conclusions are either a subformula of the major premiss ($\vee E_1$, $\vee E_2$, $\to E$) or a formula that cannot be further decomposed with top-down rules ($\to E_\vee$).

5.3 The Refutational Case

The rules of $\overline{\mathbf{N}}^{*}_{BOX}$ are quite similar to its assertional counterpart, beginning also with the ones that bring assumptions into top-down reasoning and top-down conclusions into bottom-up reasoning.

$$\frac{F^k \in \Gamma}{\Gamma \dashv F\downarrow}\ Asm \qquad\qquad \frac{\Gamma \dashv F\downarrow}{\Gamma \dashv F; \Delta; \emptyset\uparrow}\ {\downarrow\uparrow}$$

However, there are already some oddities apparent in these two basic rules. The assumptions in Γ are now also labeled with a unique integer $k \in \{1, \ldots, |\Gamma|\}$, representing the order in which they are nested, with $k = 1$ the outermost assumption. We refer to the elements of a set structured this way as *depth-labeled formulas*, with k the *depth*. In the bottom-up sequent, we can also see a wholly new component, which we will refer to as Θ. This is a similar set consisting of elements P^k, but P is specifically a propositional constant or its negation and $k \geq 1$ is an integer that, unlike with Γ, can appear multiple times as a label in the same set.

Both of these changes are owed to a unique third source of information we must consider in the complementary calculus, that being the atomic rules.

$$\frac{}{\dashv p; \Delta; \emptyset\uparrow}\ At_{\emptyset} \qquad\qquad \frac{}{\dashv \neg p; \Delta; \emptyset\uparrow}\ \neg At_{\emptyset}$$

$$\frac{\Gamma \neq \emptyset \quad f_\Gamma(p) > 0}{\Gamma \dashv p; \Delta; p^{f_\Gamma(p)}\uparrow}\ At \qquad\qquad \frac{\Gamma \neq \emptyset \quad f_\Gamma(p) > 0}{\Gamma \dashv \neg p; \Delta; \neg p^{f_\Gamma(p)}\uparrow}\ \neg At$$

These two rules have been split into four, one pair for outside the scope of all assumptions and one we have to use once assumptions are present. All of them are bottom-up rules, since rules with the ability to quite literally introduce formulas from nothing are best treated like introduction rules.

In the absence of assumptions and therefore operative formulas, $At_{\emptyset}$ and $\neg At_{\emptyset}$ can be applied whenever we want without having to care about any other elements of the context. With assumptions, however, At and $\neg At$ have to respect the side condition that only fresh propositional constants can be assigned through them, which requires the help of the new component Θ and a function $f_\Gamma(p)$ to keep track of what has already been used.

Definition 17. Let p be a propositional constant, Γ a set of depth-labeled formulas, and $\Gamma_p = \{F^k \in \Gamma \mid p \text{ occurs in } F\}$. Then

$$f_\Gamma(p) = \begin{cases} n-1 & \text{if } F^n \in \Gamma_p, \text{ and } G^m \notin \Gamma_p \text{ for } m < n \\ |\Gamma| & \text{otherwise} \end{cases}$$

□

This function tells us the deepest level of our nested assumptions we can go into before encountering a given propositional constant, being 0 if p already appears in the outermost assumption scope, and equal to the total number of assumptions if it is truly fresh. This gives us the ideal place to apply atomic rules without being blocked by an assumption, but does not yet address the main difficulty that arises when moving from a linear boxed notation to a tree notation: Operative formulas that could previously be discerned simply by checking the lines above the current one may now also be found in neighboring branches, and the information on which atomic rule applications they would hinder needs to be transferred through the derivation. This purpose is served by the Θ component, where formulas introduced by At and $\neg At$ are collected and labeled with the assumption depth at which we want to introduce them, making it possible to check for conflicts in rules with multiple premisses. A welcome simplification is that, due to *Triv* allowing unrestricted reuse of previously derived formulas in $\overline{\mathbf{N}}_{BOX}$, we can always apply the atomic rules at the very top of the assumption scope we want them in, meaning any operative formulas not introduced by assumptions (which obviously need to come first) or other atomic rule applications (which would mutually block each other no matter the order) can be safely ignored.

$$\frac{}{\Gamma \dashv \bot; \Delta; \emptyset \uparrow}\ \bot I \qquad \frac{\bot^1 \dashv F; \emptyset; \Theta \uparrow}{\dashv F; \Delta; \Theta \uparrow}\ BOX$$

$$\frac{\Gamma \dashv A \vee B \downarrow}{\Gamma \dashv A \downarrow}\ \vee E_1 \qquad \frac{\Gamma \dashv A \vee B \downarrow}{\Gamma \dashv B \downarrow}\ \vee E_2 \qquad \frac{\Gamma \dashv A \to B \downarrow}{\Gamma \dashv \neg A \downarrow}\ \to E_1 \qquad \frac{\Gamma \dashv A \to B \downarrow}{\Gamma \dashv B \downarrow}\ \to E_2$$

$$\frac{\Gamma \dashv A; \Delta; \Theta \uparrow}{\Gamma \dashv A \wedge B; \Delta; \Theta \uparrow}\ \wedge I_1 \qquad \frac{\Gamma \dashv B; \Delta; \Theta \uparrow}{\Gamma \dashv A \wedge B; \Delta; \Theta \uparrow}\ \wedge I_2$$

$$\frac{\Gamma \dashv \top; \langle \top E \rangle, \Delta; \Theta \uparrow \quad F \neq \top \quad \langle \top E \rangle \notin \Delta}{\Gamma \dashv F; \Delta; \Theta \uparrow}\ \top E$$

These rules can be brought over from the original system without any significant changes.

As previously mentioned, Θ becomes very relevant in the case of rules that require multiple bottom-up premisses, whose derivations can independently use atomic rules that may end up being in conflict.

$$\frac{\Gamma \dashv A; \Delta; \Theta_1 \uparrow \quad A^{|\Gamma|+1}, \Gamma \dashv B; \emptyset; \Theta_2 \uparrow \quad A \notin \Gamma \quad BrCon(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \vee B; \Delta; \Theta_1 \cup \Theta_2 \uparrow}\ \vee I_1$$

$$\frac{\Gamma \dashv B; \Delta; \Theta_1 \uparrow \quad B^{|\Gamma|+1}\Gamma \dashv A; \emptyset; \Theta_2 \uparrow \quad B \notin \Gamma \quad BrCon(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \vee B; \Delta; \Theta_1 \cup \Theta_2 \uparrow}\ \vee I_2$$

$$\frac{\Gamma \dashv A; \Delta; \Theta_1 \uparrow \quad \Gamma \dashv B; \Delta; \Theta_2 \uparrow \quad A \in \Gamma \text{ or } B \in \Gamma \quad BrCon(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \vee B; \Delta; \Theta_1 \cup \Theta_2 \uparrow}\ \vee I^*$$

$$\frac{\Gamma \dashv \neg A; \Delta; \Theta_1 \uparrow \quad \neg A^{|\Gamma|+1}, \Gamma \dashv B; \emptyset; \Theta_2 \uparrow \quad \neg A \notin \Gamma \quad \mathit{BrCon}(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \to B; \Delta; \Theta_1 \cup \Theta_2 \uparrow} \to I_1$$

$$\frac{\Gamma \dashv B; \Delta; \Theta_1 \uparrow \quad B^{|\Gamma|+1}, \Gamma \dashv \neg A; \emptyset; \Theta_2 \uparrow \quad B \notin \Gamma \quad \mathit{BrCon}(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \to B; \Delta; \Theta_1 \cup \Theta_2 \uparrow} \to I_2$$

$$\frac{\Gamma \dashv \neg A; \Delta; \Theta_1 \uparrow \quad \Gamma \dashv B; \Delta; \Theta_2 \uparrow \quad \neg A \in \Gamma \text{ or } B \in \Gamma \quad \mathit{BrCon}(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \to B; \Delta; \Theta_1 \cup \Theta_2 \uparrow} \to I^*$$

$\mathit{BrCon}(\Theta_1, \Theta_2, n)$ denotes the condition that there may be no pair of elements $p^j \in \Theta_1, \neg p^k \in \Theta_2$ or $\neg p^j \in \Theta_1, p^k \in \Theta_2$ such that $j \leq n$ and/or $k \leq n$. Thus, the condition $\mathit{BrCon}(\Theta_1, \Theta_2, |\Gamma|)$ makes it so these branching rules can only be applied if there are no contradictory atomic rule applications in either branch (unless they are hidden in deeper assumption scopes on both sides), and $\Theta_1 \cup \Theta_2$ in the conclusion ensures all recorded atomic rule applications from both branches are passed on to subsequent rules.

The conditions of the form $A \notin \Gamma$, here used as a shorthand for $A^n \notin \Gamma$ (with any n), are to avoid repeated assumptions that could lead to infinite loops. However, we still want to be able to introduce composite formulas with $\vee$ and $\to$ even if a required assumption is already present, which is precisely what the additional variants $\vee I^*$ and $\to I^*$ enable us to do.

When it comes to branching rules, we obviously cannot forget $\wedge E$ and its total of three different branches. With $\Gamma_A = A^{|\Gamma|+1}, \Gamma$ and $\Gamma_B = B^{|\Gamma|+1}, \Gamma$, we have

$$\frac{\Gamma \dashv A \wedge B \downarrow \quad \Gamma_A \dashv F; \emptyset; \Theta_1 \quad \Gamma_B \dashv F; \emptyset; \Theta_2 \quad A \notin \Gamma \quad B \notin \Gamma \quad \mathit{BrCon}(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv F; \Delta; \Theta_1 \cup \Theta_2 \uparrow} \wedge E.$$

Both minor premisses can have independent atomic rule applications within their newly introduced subdeduction, but any that are instead placed in containing scopes must be consistent with the other side. The major premiss, being represented by a top-down sequent, cannot have any atomic rules in its derivation and is therefore irrelevant for this condition.

The contradiction rules also introduce assumptions, but only along a single branch.

$$\frac{\neg F^{|\Gamma|+1}, \Gamma \dashv \top; \emptyset; \Theta \uparrow \quad F \neq \top \quad F \neq \neg F' \quad \neg F \notin \Gamma}{\Gamma \dashv F; \Delta; \Theta \uparrow} \mathit{CRV}$$

$$\frac{F^{|\Gamma|+1}, \Gamma \dashv \top; \emptyset; \Theta \uparrow \quad F \notin \Gamma}{\Gamma \dashv \neg F; \Delta; \Theta \uparrow} \neg I$$

Θ retains any elements from the subdeduction even after discharging the assumption, since a neighboring branch could still contain a conflicting atomic rule application in a shared assumption scope.

Finally, the $\neg E$ rule contains the pitfall of being the only elimination rule whose major premiss can be obtained directly by application of an atomic rule. Normal derivations do

not guard us against such a construction, and indeed the whole idea behind normalisation—that elimination rules only produce information that must already be present if their premisses can be derived with introduction rules—is effectively broken by having a rule able to introduce the premiss from zero prior information. Therefore, $\overline{\mathbf{N}}^{*}_{BOX}$ will have to go out of its way to accommodate this not-quite-normal construction with a secondary variation of the rule:

$$\frac{\Gamma \dashv \neg F \downarrow \quad \Gamma \dashv F; \langle \neg E \rangle, \Delta; \Theta \uparrow \quad \Gamma \neq \emptyset \quad \langle \neg E \rangle \notin \Delta}{\Gamma \dashv \top; \Delta; \Theta \uparrow} \neg E$$

$$\frac{\dfrac{}{\Gamma \dashv \neg p; \Delta; \neg p^{k} \uparrow} \neg At \qquad \Gamma \dashv p \downarrow \quad \Gamma \neq \emptyset}{\Gamma \dashv \top; \Delta; \neg p^{k} \uparrow} \neg E_{At}$$

If the major premiss is a bottom-up sequent derived specifically by $\neg At$ (it cannot be $\neg At_{\emptyset}$, since assumptions must be present to apply either $\neg E$), the formula is a negated propositional constant $\neg p$. The minor premiss must then be exactly p, which conveniently allows limiting its derivation to top-down reasoning only. This, in turn, lets us forgo use of the Δ component for loop avoidance in this variation of the rule, since neither branch can contain another copy of the conclusion sequent.

To better understand how the Θ component impacts proof search, let us consider a brief example. Tamminga [39] himself gives this derivation of $\dashv ((p \to q) \wedge \neg p) \to \neg q$ in his original work:

1.	$\neg q$	$\neg At$
2.	$\neg q$	assumption
3.	p	At
4.	$(p \to q) \wedge \neg p$	assumption
5.	$p \to q$	assumption
6.	q	$\to E_2$: 5
7.	$\top$	$\neg E$: 2, 6
8.	$\neg p$	assumption
9.	p	*Triv*: 3
10.	$\top$	$\neg E$: 8, 9
11.	$\top$	$\wedge E$: 4, 5–7, 8–10
12.	$\neg((p \to q) \wedge \neg q)$	$\neg I$: 4–11
13.	$((p \to q) \wedge \neg p) \to \neg q$	$\to I_2$: 1, 2–12

Aside from the initial applications of atomic rules at lines 1 and 3, a key point here is that the *Triv* application at line 9 repeats a conclusion of one such inference in a deeper assumption scope, where it could not have been made directly due to $(p \to q) \wedge \neg p$ being

operative. The fact that the atomic rules can become blocked by additional assumptions and necessitate constructions like this one is the main source of extra complexity in $\overline{\mathbf{N}}^*_{BOX}$.

Now, a program trying to derive $\dashv ((p \to q) \wedge \neg p) \to \neg q$ in $\overline{\mathbf{N}}^*_{BOX}$ would begin by looking for rules it can apply to this formula in bottom-up direction—to speed things up, let us say $\to I_2$ is selected for the first attempt. We obtain the following partial derivation:

$$\frac{\dashv \neg q; \emptyset; \Theta_1 \uparrow \quad \neg q^1 \dashv \neg((p \to q) \wedge \neg p); \emptyset; \Theta_2 \uparrow \quad BrCon(\Theta_1, \Theta_2, 0)}{\dashv ((p \to q) \wedge \neg p) \to \neg q; \emptyset; \Theta_1 \cup \Theta_2 \uparrow} \to I_2$$

Note that $BrCon(\Theta_1, \Theta_2, 0)$ is trivially fulfilled because neither Θ set can contain a formula with a depth label of 0 or below.

In the next step, we look for a way to derive the first premiss, and find:

$$\frac{\dfrac{}{\dashv \neg q; \emptyset; \emptyset \uparrow}\, \neg At_\emptyset \quad \neg q^1 \dashv \neg((p \to q) \wedge \neg p); \emptyset; \Theta_2 \uparrow}{\dashv ((p \to q) \wedge \neg p) \to \neg q; \emptyset; \Theta_2 \uparrow} \to I_2$$

We now know that $\Theta_1 = \emptyset$, and Θ_2 consists of zero or more yet unknown elements at a depth of at least 1.

What remains is the second premiss, which we tackle with the next inference step:

$$\frac{\dfrac{\dfrac{}{\dashv \neg q; \emptyset; \emptyset \uparrow}\, \neg At_\emptyset \quad \dfrac{((p \to q) \wedge \neg p)^2, \neg q^1 \dashv \top; \emptyset; \Theta_2 \uparrow}{\neg q^1 \dashv \neg((p \to q) \wedge \neg p); \emptyset; \Theta_2 \uparrow} \neg I}{\dashv ((p \to q) \wedge \neg p) \to \neg q; \emptyset; \Theta_2 \uparrow} \to I_2$$

To save space, we have omitted the evidently true side conditions of $\neg I$ and $\to I_2$.

We could try to use $\neg E$ on the assumption $(\neg q, 1)$, but since q already appears in that very assumption, neither At nor any other rule can be used to derive it. Instead, we define Γ to be the set of assumptions $\{((p \to q) \wedge \neg p)^2, \neg q^1\}$ and apply $\wedge E$ to construct the following derivation δ (with the side condition $BrCon(\Theta_3, \Theta_4, 2)$ omitted for brevity):

$$\frac{\dfrac{((p \to q) \wedge \neg p)^2 \in \Gamma}{\Gamma \dashv (p \to q) \wedge \neg p \downarrow}\, Asm \quad (p \to q)^3, \Gamma \dashv \top; \emptyset; \Theta_3 \uparrow \quad \neg p^3, \Gamma \dashv \top; \emptyset; \Theta_4 \uparrow}{\Gamma \dashv \top; \emptyset; \Theta_2 = (\Theta_3 \cup \Theta_4) \uparrow} \wedge E$$

Which makes the whole derivation

$$\frac{\dfrac{\dfrac{}{\dashv \neg q; \emptyset; \emptyset \uparrow}\, \neg At_\emptyset \quad \dfrac{\delta}{\neg q^1 \dashv \neg((p \to q) \wedge \neg p); \emptyset; \Theta_2 \uparrow} \neg I}{\dashv ((p \to q) \wedge \neg p) \to \neg q; \emptyset; \Theta_2 \uparrow} \to I_2.$$

The two branches of bottom-up reasoning in δ each represent a distinct subdeduction with its own third assumption, and each of them can contain different elements with depth exactly 3 in their respective Θ component.

All that is left is to derive $\top$, first in the left branch as δ_1:

$$\dfrac{\dfrac{\neg q^1 \in (p \to q)^3, \Gamma}{(p \to q)^3, \Gamma \dashv \neg q \downarrow}\, Asm \quad \dfrac{\dfrac{\dfrac{(p \to q)^3 \in (p \to q)^3, \Gamma}{(p \to q)^3, \Gamma \dashv p \to q \downarrow}\, Asm}{(p \to q)^3, \Gamma \dashv q \downarrow}\, \to E_2}{(p \to q)^3, \Gamma \dashv q; \langle \neg E \rangle; \emptyset \uparrow}\, \downarrow\uparrow}{(p \to q)^3, \Gamma \dashv \top; \emptyset; \Theta_3 = \emptyset \uparrow}\, \neg E$$

In view of the $\downarrow\uparrow$ rule, we obtain the information that $\Theta_3 = \emptyset$.

The right branch requires the following δ_2:

$$\dfrac{\dfrac{\neg p^3 \in \neg p^3, \Gamma}{\neg p^3, \Gamma \dashv \neg p \downarrow}\, Asm \quad \dfrac{\neg p^3, \Gamma \neq \emptyset \quad f_{(\neg p^3, \Gamma)}(p) = 1 > 0}{\neg p^3, \Gamma \dashv p; \langle \neg E \rangle; p^1 \uparrow}\, At}{\neg p^3, \Gamma \dashv \top; \emptyset; \Theta_4 = \{p^1\} \uparrow}\, \neg E$$

Putting everything together, we have finally completed the derivation.

$$\dfrac{\dfrac{}{\dashv \neg q; \emptyset; \emptyset \uparrow}\, \neg At_\emptyset \quad \dfrac{\dfrac{\dfrac{\dfrac{((p \to q) \wedge \neg p)^2 \in \Gamma}{\Gamma \dashv (p \to q) \wedge \neg p \downarrow}\, Asm \quad \delta_1 \quad \delta_2}{\Gamma \dashv \top; \emptyset; \Theta_2 = (\emptyset \cup \Theta_4) \uparrow}\, \wedge E}{\neg q^1 \dashv \neg((p \to q) \wedge \neg p); \emptyset; \Theta_2 \uparrow}\, \neg I}{\dashv ((p \to q) \wedge \neg p) \to \neg q; \emptyset; \Theta_2 \uparrow}\, \to I_2$$

Since Θ_4 has now been specified exactly, figuring out what is contained in the remaining Θ sets is merely a matter of propagating that information down the derivation.

$$\begin{aligned} \Theta_4 &= \{p^1\} \\ \Theta_3 &= \emptyset \\ \Theta_2 &= \emptyset \cup \{p^1\} = \{p^1\} \\ \Theta_1 &= \emptyset \end{aligned}$$

Therefore, the $\wedge E$ side condition $BrCon(\Theta_3, \Theta_4, 2)$ means $BrCon(\emptyset, \{p^1\}, 2)$ and is clearly fulfilled.

5.3.1 Soundness

Soundness of $\overline{\mathbf{N}}^*_{BOX}$ follows from soundness of $\overline{\mathbf{N}}_{BOX}$ if it is possible to translate all derivations from the former system into the latter.

Theorem 9. *For any derivation in $\overline{\mathbf{N}}^*_{BOX}$ with root sequent $\Gamma \dashv F; \Delta; \Theta \uparrow$ or $\Gamma \dashv F \downarrow$, there exists a derivation in $\overline{\mathbf{N}}_{BOX}$ whose last line contains the formula F with active assumptions Γ.*

Proof. Because the handling of operative formulas and the atomic rules is so different, this translation is significantly less obvious than it was for the assertional system. We will show that our conditions are sufficient to emulate those from $\overline{\mathbf{N}}_{BOX}$, by induction on the depth d of the given derivation.

Base Case: $d = 1$. Thus, the root rule is one of $\bot I$, Asm, $At_\emptyset$, $\neg At_\emptyset$, At, or $\neg At$.

- If $\bot I$, we can directly use the corresponding rule of $\overline{\mathbf{N}}_{BOX}$, embedded in the assumptions from Γ (if any).
- If Asm, we introduce the assumptions from Γ and use $Triv$ to place the desired one on the final line.
- If $At_\emptyset$ or $\neg At_\emptyset$, then $\Gamma = \emptyset$, no formulas are operative by definition, and the At or $\neg At$ rule of $\overline{\mathbf{N}}_{BOX}$ can be freely applied to derive the conclusion we want.
- If At or $\neg At$, then $F = p$ or $F = \neg p$ and there exists an integer $0 < f_\Gamma(p) \leq |\Gamma|$. After introducing exactly that number of assumptions (in ascending order of their labeled depth), the corresponding rule of $\overline{\mathbf{N}}_{BOX}$ must still be applicable, because if those assumptions included a formula F containing p, we would have $F^n \in \Gamma_p$ for some $n < f_\Gamma(p)$, which would imply $f_\Gamma(p) \leq n - 1$ and thus be contradictory. After introducing the remaining assumptions, $Triv$ can be used to get F on the final line.

Induction Hypothesis: For any derivation in $\overline{\mathbf{N}}^*_{BOX}$ with depth $1 \leq d \leq n - 1$ with root sequent $\Gamma \dashv F; \Delta; \Theta \uparrow$ or $\Gamma \dashv F \downarrow$, there exists a derivation in $\overline{\mathbf{N}}_{BOX}$ whose last line contains the formula F with active assumptions Γ. Also, we can make the following statements about applications of the atomic rules in the $\overline{\mathbf{N}}_{BOX}$ derivation:

- For each occurrence of At or $\neg At$ outside the scope of all assumptions, there is an application of $At_\emptyset$ or $\neg At_\emptyset$, respectively, in the $\overline{\mathbf{N}}^*_{BOX}$ derivation.
- For each occurrence of At within the scope of one or more assumptions, there is an application of At in the $\overline{\mathbf{N}}^*_{BOX}$ derivation introducing a sequent $\Gamma^* \dashv p; \Delta^*; p^n \uparrow$. Γ^* is a set of assumptions including, but not limited to, all assumptions active at the point of the At application in the $\overline{\mathbf{N}}_{BOX}$ derivation, p the introduced propositional constant, and n exactly the number of assumptions in whose scope the line with At is. Δ^* represents the loop avoidance component, with the stipulation that $\Delta \subseteq \Delta^*$ if $\Gamma^* = \Gamma$.
- For each occurrence of $\neg At$ within the scope of one or more assumptions, there is an application of $\neg At$ in the $\overline{\mathbf{N}}^*_{BOX}$ derivation introducing a sequent $\Gamma^* \dashv \neg p; \Delta^*; \neg p^n \uparrow$, with the symbols defined as above.

Induction Step: We make a case distinction at the root rule.

- $\downarrow\uparrow$: From the premiss $\Gamma \dashv F \downarrow$, we have by the induction hypothesis a derivation in $\overline{\mathbf{N}}_{BOX}$ ending on formula F with assumptions Γ, which is just what we need.
- BOX: The derivation of F under the assumption $\bot$ we get from the induction hypothesis can immediately be extended with the BOX rule of $\overline{\mathbf{N}}_{BOX}$.
- $\wedge E_1$, $\wedge E_2$, $\rightarrow E_1$, $\rightarrow E_2$, $\wedge I_1$, $\wedge I_2$, $\top E$, CRV, $\neg I$: For all these rules, we have a derivation ending precisely on the one premiss needed to apply them.
- $\vee I_1$: By the induction hypothesis, we have a derivation in $\overline{\mathbf{N}}_{BOX}$ of A in the scope of the assumptions Γ, as well as one of B in the scope of assumptions A, Γ. In order to extend these derivations to one of $A \vee B$ via $\vee I_1$, we must place the subdeduction concluding B below the line where A is derived. If $\Gamma = \emptyset$, doing so is easy, but otherwise, the potential existence of atomic rules at any assumption level of either derivation makes some careful interleaving necessary. Specifically, we take each line concluded by At or $\neg At$ in the derivation of B, except those within the scope of the innermost assumption A, and insert it into the derivation of A so it stands within the scope of exactly as many assumptions as before, and after any atomic rule applications already present (duplicates are simply skipped if they are at the same depth, otherwise the deeper one is replaced by $Triv$).

  We can be sure doing so will not produce any conflicts, because by the induction hypothesis, each line with At or $\neg At$ in the derivation of A (within the scope of at least one assumption) corresponds to a sequent $\Gamma^* \dashv p; \Delta^*; p^n \uparrow$ or $\Gamma^* \dashv \neg p; \Delta^*; \neg p^n \uparrow$ derived by At or $\neg At$, respectively, and the same is true in the derivation of B with an additional assumption A. If there was, for instance, both an At application in the derivation of A concluding p at depth j and a $\neg At$ application in the derivation of B concluding $\neg p$ at depth k, we would have $p^j \in \Theta_1$, $p^k \in \Theta_2$. In this scenario, the only way to satisfy $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ is if $j, k > |\Gamma|$, which translated to $\overline{\mathbf{N}}_{BOX}$ means the At application occurs in some subdeduction that is closed before the conclusion A and p is no longer operative at that point, while the $\neg At$ application occurs after the assumption A immediately following that conclusion. Therefore, deriving $\neg p$ by $\neg At$ is completely legal, and the same reasoning applies if At and $\neg At$ are switched.

  It follows that, given that $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ must indeed be satisfied in the derivation we are translating, interleaving the two given derivations in $\overline{\mathbf{N}}_{BOX}$ as described is a safe process, and once we have done so, $\vee I_1$ can be applied at the line immediately following the subdeduction.
- $\vee I_2$, $\rightarrow I_1$, $\rightarrow I_2$ all follow the same logic as $\vee I_1$.
- $\vee I^*$: Things are slightly different in this case, as the induction hypothesis only gives us derivations for A and B in the scope of Γ, while $\overline{\mathbf{N}}_{BOX}$ needs one of them to have the other formula as an additional assumption. However, we also know that $A \in \Gamma$ or $B \in \Gamma$, so introducing that already present assumption a second time will not lead to any new operative formulas that could interfere with atomic rules

and we can add it to the respective derivation with no issues. This then gives us two derivations we can interleave and extend as we normally would for $\vee I_1$ or $\vee I_2$.

- $\rightarrow I^*$: Works as above.
- $\wedge E$: We have a derivation in $\overline{\mathbf{N}}_{BOX}$ of $A \wedge B$ in the scope of Γ, and two of F with an additional assumption A or B, respectively. Since no atomic rules can be used in the derivation of a top-down sequent, we know by the induction hypothesis that the derivation for the major premiss also contains none in its converted form. We can therefore easily merge the derivation of F with the extra assumption A into this derivation, placing all lines with At or $\neg At$ right at the start of the appropriate assumption scope, and then add the derivation of F with the extra assumption B by the same interleaving tactic used before. The end result is the exact configuration of lines needed to apply $\wedge E$.
- $\neg E$, $\neg E_{At}$: Since one of the premisses is always a top-down sequent and thus translates to a derivation that can have no atomic rules in it, merging it with the other branch is a simple matter. All that needs to be done from there is adding a line with $\top$ derived by $\neg E$ at the end.

□

5.3.2 Completeness

The Γ-weakening lemma can be applied to $\overline{\mathbf{N}}^*_{BOX}$ as well, though in a slightly altered form to account for the special nature of derivations outside the scope of all assumptions.

Lemma 10. *Given a derivation in $\overline{\mathbf{N}}^*_{BOX}$ with assumptions $\Gamma \neq \emptyset$ and formula F at the root, and a set of depth-labeled formulas $\Gamma' \supseteq \Gamma$, there exists a derivation in $\overline{\mathbf{N}}^*_{BOX}$ with assumptions Γ' and formula F at the root.*

Proof. By induction on the depth d of the given derivation.

Base Case: $d = 1$. Thus the root rule is one of $\bot I$, *Asm*, *At*, or $\neg At$. The first two have no conditions that could be hindered by the presence of additional assumptions, but the atomic rules are not quite as obvious. If, for example, $f_\Gamma(p) > 0$ and $f_{\Gamma'}(p) = 0$, we would have the issue that At can no longer be applied to derive p after exchanging Γ for Γ'. However, this can be ruled out because $f_{\Gamma'}(p) = 0$ would require the existence of a formula $P^1 \in \Gamma'$ containing p, while $f_\Gamma(p) > 0$ simultaneously requires $P^1 \notin \Gamma$. But the depth labels on the assumptions are by definition unique and cover exactly the integers $\{1, \ldots, |\Gamma|\}$, so there must already be $Q^1 \in \Gamma$ so that Q does not contain p. Obviously, $P \neq Q$, and so we cannot have $P^1 \in \Gamma'$, as it would be a duplicate depth label. Therefore, atomic rules can still be applied even with additional assumptions in place.

Induction Hypothesis: For any derivation in $\overline{\mathbf{N}}^*_{BOX}$ with depth $1 < d \leq n - 1$ that has assumptions $\Gamma \neq \emptyset$ and formula F at the root, there exists a derivation in $\overline{\mathbf{N}}^*_{BOX}$ with assumptions Γ' and formula F at the root, where $\Gamma' \supset \Gamma$.

Induction Step: We distinguish by the root rule.

- *CRV*, $\neg I$: If an assumption is present that violates the side condition, we have $\Gamma' \dashv \top; \Delta; \Theta \uparrow$ by the induction hypothesis and contraction, which lets us proceed by $\top E$ instead.
- $\wedge E$: If $A \in \Gamma'$, we have, by contraction on $A^{|\Gamma'|+1}, \Gamma' \dashv F; \Delta; \Theta \uparrow$ the sequent $\Gamma' \dashv F; \Delta; \Theta^{|\Gamma'|} \uparrow$, immediately resolving the issue. The same approach works for $B \in \Gamma'$, and any other case simply does not violate side conditions in the first place.
- $\vee I_1$, $\vee I_2$, $\to I_1$, $\to I_2$: If the original rule is blocked by $A \in \Gamma'$ (for the $\vee I$ rules), $\neg A \in \Gamma'$ (for the $\to I$) rules, or $B \in \Gamma'$, it also means the side conditions of $\vee I^*$ or $\to I^*$ are fulfilled, so we can use the appropriate one out of those rules instead.
- No other rules (that have sequents as premisses) feature conditions that could conflict with exchanging Γ for Γ'.

□

Theorem 10. *For any (simplified and direct) normal derivation in $\overline{\mathbf{N}}_{BOX}$ whose last line contains the formula F with active assumptions Γ, there exists a derivation in $\overline{\mathbf{N}}^*_{BOX}$ with root sequent $\Gamma \vdash F; \emptyset; \Theta \uparrow$.*

Proof. We proceed by induction on the length l of the $\overline{\mathbf{N}}_{BOX}$ derivation.

Base Case: $l = 1$. Then, the formula F can be an assumption, $\bot$ derived by $\bot I$, or a propositional constant p or its negation $\neg p$ derived by an atomic rule. The $\overline{\mathbf{N}}^*_{BOX}$ derivations for case one and two mirror the ones we already saw in the assertional system:

$$\dfrac{\dfrac{F^n \in \Gamma}{\Gamma \vdash F \downarrow}\, Asm}{\Gamma \vdash F; \emptyset; \emptyset \uparrow} \downarrow\uparrow \qquad\qquad \dfrac{}{\Gamma \vdash \bot; \emptyset; \emptyset \uparrow}\, \bot I$$

For the atomic rules, we know that the derivation consists only of a single line and does not have any space for assumptions before its conclusion, so $\Gamma = \emptyset$ and we can simply use the rules designed specifically for that case.

$$\dfrac{}{\vdash p; \emptyset; \emptyset \uparrow}\, At_\emptyset \qquad\qquad \dfrac{}{\vdash \neg p; \emptyset; \emptyset \uparrow}\, \neg At_\emptyset$$

Induction Hypothesis: For a normal derivation in $\overline{\mathbf{N}}_{BOX}$ with length $1 < l \leq n-1$ that has formula F and active assumptions Γ at line l, there exists a derivation in $\overline{\mathbf{N}}^*_{BOX}$ with root sequent $\Gamma \vdash F; \emptyset; \Theta \uparrow$. If the original derivation ended on an elimination rule, the root rule is $\downarrow\uparrow$, $\wedge E$ (if $F = \top$), or $\top E$.

Induction Step: We distinguish by the final rule applied on line n of the derivation in $\overline{\mathbf{N}}_{BOX}$.

- *Triv*: We obtain directly from the induction hypothesis a $\overline{\mathbf{N}}^*_{BOX}$ derivation $\Gamma' \dashv F; \emptyset; \Theta \uparrow$, with $\Gamma' \neq \emptyset$ and $\Gamma' \subseteq \Gamma$. This immediately gives us $\Gamma \dashv F; \emptyset; \Theta \uparrow$ by Γ-weakening.

- *BOX*: The induction hypothesis gives us a derivation ending on $\bot^1 \dashv F; \emptyset; \Theta \uparrow$, the exact sequent needed to apply our own version of the *BOX* rule and complete the derivation outside the scope of all assumptions.

- $\bot I$: No different from the base case.

- $At, \neg At$: If $\Gamma = \emptyset$, we can use $At_\emptyset$ or $\neg At_\emptyset$ just like in the base case. Otherwise, we know no assumption in Γ contains the propositional constant p ($F = p$ or $F = \neg p$), so it is safe to use the appropriate one of the following two derivations:

$$\frac{}{\Gamma \vdash p; \emptyset; p^{|\Gamma|} \uparrow}\ At \qquad\qquad \frac{}{\Gamma \vdash \neg p; \emptyset; \neg p^{|\Gamma|} \uparrow}\ \neg At$$

- $\vee E_1$, $\vee E_2$, $\to E_1$, $\to E_2$: We are working with normal derivations, so the premiss must have originally been derived by an elimination rule, giving us by the induction hypothesis a derivation with root rule $\downarrow\uparrow$ or $\top E$. In the first case, we have the top-down sequent we need as our sole premiss and can use $\downarrow\uparrow$ to get a bottom-up sequent again after applying the appropriate rule, and in the latter case all we need to do is replace the conclusion of $\top E$.

- $\wedge I_1$, $\wedge I_2$: As there are no special side conditions, we can always use the appropriate rule in $\overline{\mathbf{N}}^*_{BOX}$ as well.

- $\top E$: If $\top E$ is not used under the exact assumptions Γ in the derivation of $\Gamma \dashv \top; \emptyset; \Theta \uparrow$ we have by the induction hypothesis, we can simply add $\langle \top E \rangle$ to the Δ component and use that sequent as our premiss. Otherwise, the conflicting rule application must already have such a sequent as its premiss, and we can construct our derivation of F by replacing the conclusion.

- $\neg I$: We have access to the bottom-up sequent $F^{|\Gamma|+1}, \Gamma \dashv \top; \emptyset; \Theta' \uparrow$ by the induction hypothesis, which is all we need if $F \notin \Gamma$. Otherwise, we have the premiss $\Gamma \dashv \top; \emptyset; \Theta \uparrow$ for $\top E$ by contraction and can simply use that rule instead.

- *CRV*: Violations of the side condition $\neg F \notin \Gamma$ can be handled as above. And unlike in the assertional system, the lack of a top-down rule that can use new assumptions to extract additional information from older assumption means there are not even any cases where subdeduction assumptions $\neg\top$ or $\neg\neg F'$ would falsely appear to be useful.

- $\neg E$: If the major premiss $\neg F$ was derived by an elimination rule, the induction hypothesis gives us either $\Gamma \dashv \top; \emptyset; \Theta \uparrow$, which is already the desired conclusion, or $\Gamma \dashv \neg F \downarrow$, which is the exact sequent needed for the major premiss of $\neg E$ in $\overline{\mathbf{N}}^*_{BOX}$. In the latter case, we also have $\Gamma \dashv F; \emptyset; \Theta$ for the minor premiss, but strictly

speaking need $\Gamma \dashv F; \langle \neg E \rangle; \Theta$. Any additional application of $\neg E$ that would cause a conflict in the derivation of this sequent would have as its conclusion $\Gamma \dashv \top; \emptyset; \Theta \uparrow$, which, again, is exactly what we need.

If the major premiss was derived by a non-elimination rule, that rule can only be $\neg At$ in a normal derivation, and F is a propositional constant p. In that case, we can apply $\neg E_{At}$, but there must be a top-down sequent for the minor premiss p available. Assume this is not the case: Then it follows from the induction hypothesis that either the root rule is $\top E$ (enabling a trivial solution) or p was not obtained by an elimination rule in the original derivation. With At not an option due to the conflicting $\neg At$, the only way to obtain a propositional constant by a non-elimination rule would be CRV, requiring a derivation of $\top$ under the assumption $\neg p$.

Now recall that soundness in $\overline{\mathbf{N}}$, and by extension $\overline{\mathbf{N}}_{BOX}$, is defined by the statement "if $\models \mathfrak{R}_n$, then $\models \mathfrak{R}_n \cup \{F_n\}$" where F_n denotes the formula at line n and $\mathfrak{R}_n$ the set of operative formulas at that line (excluding F_n itself, unless it is being introduced as an assumption). There can be no countermodels for any set of formulas with $\top$ in it, so deriving it in a sound manner is only possible if $\mathfrak{R}_n$ also had no countermodels. But because all rules adhere to this soundness condition, the only way to add to the set of operative formulas so that $\mathfrak{R}_n \cup \{F_n\}$ has no countermodels even though $\mathfrak{R}_n$ had at least one is by introducing an assumption. Furthermore, the $\neg p$ assumption for CRV cannot have had this effect, as it was either introduced after deriving $\neg p$ by $\neg At$ and did not contribute any new information, or introduced in a state where no operative formula contained the propositional constant p, which should have allowed us to preserve any existing countermodels by simply extending them with the appropriate value for p. The only remaining possibility is that a contradiction that allows deriving $\top$ was already present in the operative formulas before the CRV subdeduction, so there is no need to use $\neg E$ with major premiss $\neg p$ in the first place.

- $\vee I_1$: For $F = A \vee B$, we have given $\Gamma \dashv A; \emptyset; \Theta_1 \uparrow$ and $A, \Gamma \dashv B; \emptyset; \Theta_2 \uparrow$. Assuming $A \notin \Gamma$, we still need to ensure $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ holds, i.e., there may not be any pair of At and $\neg At$ using the same propositional constant, unless both occur in independent assumption scopes at a depth greater than $|\Gamma|$. This is clearly true if $\Gamma = \emptyset$ since $|\Gamma| = 0$ then, but other cases must be inspected more closely. Suppose the derivation of the major premiss contains At introducing p, and the derivation of the minor premiss contains $\neg At$ introducing $\neg p$. If both happen in the scope of $|\Gamma|$ or fewer assumptions, they must have stood somewhere before the assumption A in the original derivation, and would conflict with each other no matter which order they are written. If At is outside that subdeduction and $\neg At$ is inside, there is also an obvious conflict, since p would be operative at the $\neg At$ line. However, there is also the possibility that At is applied in some arbitrary subdeduction during the derivation of the minor premiss, and $\neg At$ in the scope of $|\Gamma|$ or fewer assumptions. In such a case, the line with $\neg At$ would have to stand below that subdeduction so

as to not block At, but would itself not be blocked because p would cease to be operative before reaching it. Meanwhile, in $\overline{\mathbf{N}}^*_{BOX}$, the application of $\neg At$ would place $\neg p^k$ with $k \leq |\Gamma|$ into Θ_2, where it would conflict with $p^j \in \Theta_1$, even though $j > |\Gamma|$.

We can resolve this issue by observing that a normal derivation never has to take such a form. If At can be applied within some subdeduction in the derivation of the minor premiss, that subdeduction's assumption must be one that does not contain p, and if $\neg At$ can be applied at some point after that assumption is discharged, p can also not appear in the conclusion of the rule that discharges it, or in any assumptions in Γ. But any rule which could use p as a premiss, except for $\neg E$, is an introduction rule with a conclusion that would contain p, and in a normal derivation, the only further use for those conclusions would be introduction rules that also would not remove p. In the case of $\neg E$, it would certainly be possible that p is the minor premiss that together with $\neg p$ lets us derive $\top$, and from that a final conclusion for the subdeduction which indeed does not contain p. However, the major premiss $\neg p$ would have to be derived by elimination rules starting from some assumption containing p, which we already know are not present. So this possibility can be ruled out as well, and we can only conclude that any derivation using p introduced by At within the subdeduction does not actually connect to the final conclusion of the subdeduction, meaning it can be removed without impacting the overall derivation at all. And if we do that, we can just as well place the $\neg At$ application from after the subdeduction before it, restoring the standard form.

If At and $\neg At$ are switched in the above scenario, so that $\neg p$ is the formula we are deriving inside the additional subdeduction, the $\neg E$ case specifically is slightly different in that a normal derivation could also use introduction rules to derive the minor premiss p. However, we have previously shown that this is never actually necessary, and so there is no problem.

In summary, the only case in which $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ would not hold can always be transformed into a case where it does, and so $\vee I_1$ can be applied without issue.

For the case where $A \in \Gamma$, we have a simple solution available in the form of $\vee I^*$, since $\Gamma \dashv B; \emptyset; \Theta_2 \uparrow$ follows by contraction.

- $\vee I_2$, $\rightarrow I_1$, $\rightarrow I_2$ all follow the same structure as $\vee I_1$, and can be translated in the same way.

- $\wedge E$: The induction hypothesis gives us a sequent $\Gamma \dashv A \wedge B; \emptyset; \Theta \uparrow$ derived by $\downarrow\uparrow$ or $\top E$, as well as the sequents $A, \Gamma \dashv \top; \emptyset; \Theta_1 \uparrow$ and $B, \Gamma \dashv \top; \emptyset; \Theta_2 \uparrow$. If the bottom-up sequent for the major premiss was derived by $\top E$ we already have $\Gamma \dashv \top; \emptyset; \Theta \uparrow$, and if it was derived by $\downarrow\uparrow$ we have a matching top-down sequent. If $A \in \Gamma$ or $B \in \Gamma$, then $\Gamma \dashv \top; \emptyset; \Theta_1 \uparrow$ or $\Gamma \dashv \top; \emptyset; \Theta_2 \uparrow$ follows by contraction, so the only remaining side condition to watch out for is $BrCon(\Theta_1, \Theta_2, |\Gamma|)$. Conveniently, the top-down reasoning by which $A \wedge B$ is derived has no room for atomic rules, so the only problem can be conflicts between the two subdeduction branches. Specifically,

we need to avoid pairs of At and $\neg At$ used on the same propositional constant p in the scope of $|\Gamma|$ or fewer assumptions—what happens at the bottom level of each subdeduction does not affect the other side. Since the shape of $\wedge E$ in $\overline{\mathbf{N}}_{BOX}$ is specifically so that the two subdeductions must stand directly beneath each other, any atomic rule applications they need at shallower assumption levels must stand before them, where they would necessarily be in conflict. Thus, no problematic pair of these rules can exist and we can safely assume the condition holds.

□

5.3.3 Loop-freeness

With the $\wedge$ introduction rules so closely mirroring the $\vee$ introduction rules of the assertional system, there is of course the same kind of scenario that could potentially lead to an infinite loop in bottom-up reasoning, which the Δ component prevents in the same way:

$$\dfrac{\dfrac{\neg(p \wedge \top)^2 \in \{\neg(p \wedge \top)^2, p^1\}}{\{\neg(p \wedge \top)^2, p^1\} \vdash \neg(p \wedge \top) \downarrow}\, Asm \quad \dfrac{\dfrac{\vdots}{\{\neg(p \wedge \top)^2, p^1\} \vdash \top; \langle \neg E \rangle; \Theta \uparrow}\, \neg E?}{\{\neg(p \wedge \top)^2, p^1\} \vdash p \wedge \top; \langle \neg E \rangle; \Theta \uparrow}\, \wedge I_2}{\{\neg(p \wedge \top)^2, p^1\} \vdash \top; \emptyset; \Theta \uparrow}\, \neg E$$

Otherwise, the loop-freeness of bottom-up reasoning is apparent. $\wedge I_1$, $\wedge I_2$, $\vee I_1$, $\vee I_2$, $\rightarrow I_1$, and $\rightarrow I_2$ all have subformulas of their conclusion as premisses, BOX and $\top E$ cannot be applied in the derivation of their own premisses, and $\wedge E$, CRV, and $\neg I$ cannot be repeated with the same assumptions in the subdeductions that form their premisses. $\neg E$ is limited by Δ and thus reliant on other rules to introduce new assumptions, but this condition does not apply to $\neg E_{At}$, since its only bottom-up premiss is always derived with a single inference step.

The top-down rules are only able to extract subformulas from the given assumptions, leaving no room for any loops.

CHAPTER 6

Implementation

Our automated theorem prover Hermes implements both $\mathbf{N}^*$ and $\overline{\mathbf{N}}^*_{BOX}$, allowing it to judge a given formula either provable (i.e., valid) or refutable and produce a natural deduction derivation for this result.

6.1 Overview

6.1.1 System Architecture

Hermes is implemented using the logic programming language Prolog (more specifically, SWI-Prolog [41]), which features built-in proof search and backtracking capabilities that can be adapted for our purposes. Figure 6.1 outlines the key components of the program and their interactions.

The sole point of user interaction is the module `ndj`, which receives a formula F and outputs $\delta^{\square}_F$, a natural deduction derivation of its validity or refutability in a linear boxed style similar to the one used in $\mathbf{N}$ and $\overline{\mathbf{N}}_{BOX}$.

Internally, `ndj` concurrently calls the modules `ndprovable` and `ndrefutable`, which are the actual implementations of $\mathbf{N}^*$ and $\overline{\mathbf{N}}^*_{BOX}$, respectively. Proof search in both modules starts from a bottom-up sequent with formula F and no assumptions, and calls itself recursively with the premisses needed to derive its current goal by some appropriate rule. Each time a new assumption is introduced and a subdeduction begins during this process, the module `tdc` is used to generate the *top-down closure* $\mathtt{TDC}(\Gamma)$ from the current assumptions Γ. This is a set of derivations containing all possible conclusions that can be obtained from Γ using only top-down reasoning, which allows us to more efficiently obtain the premisses needed for $\downarrow\uparrow$ and similar rules.

Since $\mathbf{N}^*$ and $\overline{\mathbf{N}}^*_{BOX}$ are complementary as long as no assumptions are present, and both systems are loop-free, exactly one of the concurrent calls made by `ndj` will eventually

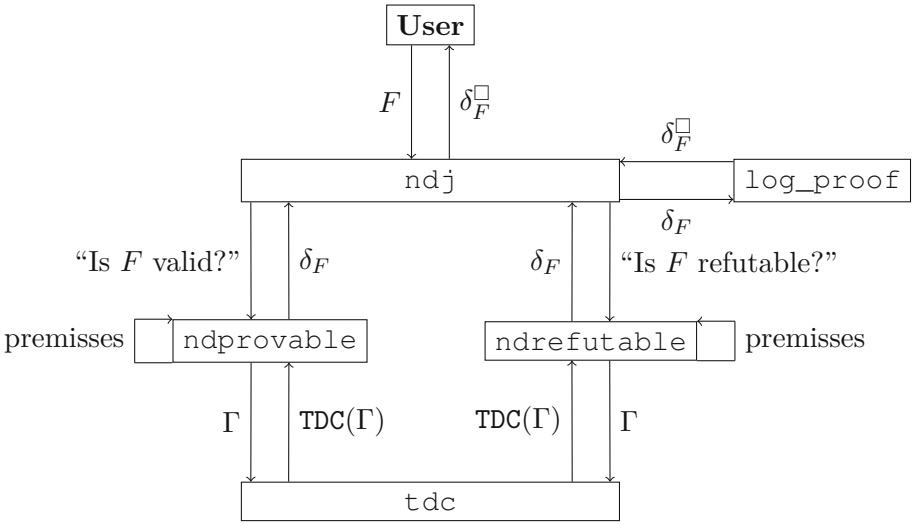


Figure 6.1: Architecture of Hermes.

succeed and yield a tree-style derivation δ_F showing either validity or refutability of F. Finally, the module `log_proof` converts δ_F to its linear representation $\delta_F^{\square}$, which is the more easily readable form we want to display to the user.

6.1.2 Input Format

The input formula F must be given as a term in prefix notation, e.g., $a \to (b \wedge c)$ would be `impl(a,and(b,c))`. The symbols of propositional logic are translated as follows:

- $A \wedge B$: `and(A,B)`;
- $A \vee B$: `or(A,B)`;
- $A \to B$: `impl(A,B)`;
- $\neg F$: `neg(F)`;
- $\top$: `1`; and
- $\bot$: `0`.

Propositional constants are denoted by arbitrary atomic terms other than `1` and `0`, usually single lowercase characters.

6.1.3 Output Format

The derivation $\delta_F^{\square}$ uses a linear boxed notation, as it is easier to display and read compared to tree-style derivations that very quickly explode in width. The notation is fairly similar to what we used for $\mathbf{N}$ and $\overline{\mathbf{N}}_{BOX}$, but has some differences to simplify conversion and display. Most notably, boxes are not drawn fully, but only hinted by zero or more copies of the character `|` written between the line number and the formula, representing the left edges of the appropriate number of nested boxes. The assumption at the start of each subdeduction is specially marked via a prefix $\star$, and is repeated via *Triv* each time it appears as the premiss of a bottom-up rule (even if the position of the assumption would also allow using it directly). We also mostly retain rules unique to $\mathbf{N}^*$ and $\overline{\mathbf{N}}^*_{BOX}$, such as the versions of $\rightarrow I$ that do not require a subdeduction if the appropriate assumption is already present. Similarly, instead of moving applications of the atomic rules so they are correctly placed with regards to operative formulas in $\overline{\mathbf{N}}_{BOX}$, we keep them at their original assumption depth and only note as a suffix of the rule name the assumption depth at which they are effectively applied (i.e., the value of $f_\Gamma(p)$ for the propositional constant p in their conclusion).

As an example, for $F = (p \wedge q) \vee p$, the program would judge the formula refutable and output the following $\delta_F^{\square}$:

```
1. | *0 [asm]
2. | p [at^1]
3. | and(p,q) [and-i1: 2.]
4. | | *and(p,q) [asm]
5. | | p [at^1]
6. | or(and(p,q),p) [or-i1: 3., 4.-5.]
7. or(and(p,q),p) [box: 1.-6.]
```

6.2 Implementational Aspects

6.2.1 Data Structures

The internal representation of formulas used in the proof search uses the same prefix notation as the input formula and the formulas contained in the output. The sets Γ, Δ, and Θ are simple Prolog lists, with Θ specifically using the *ordsets* library to avoid duplicates in $\Theta_1 \cup \Theta_2$. While duplicates in Γ and Δ are, on a technical level, possible, they are prevented by the structure of the rules in both $\mathbf{N}^*$ and $\overline{\mathbf{N}}^*_{BOX}$, which explicitly only introduce new members into these sets.

The tree-style derivation δ_F built internally is represented by the *log*, a recursive term of the form `(F,Rule,Asm,Prems)`, where `F` is the formula at the root of the derivation, `Rule` the root rule, `Asm` the assumptions at the root, and `Prems` a list containing the zero or more premisses of the root rule, each represented with a term of the same form.

A simple derivation in $\mathbf{N}^*$

$$\dfrac{\dfrac{\dfrac{p \in \{p\}}{p \vdash p \downarrow}\,Asm}{p \vdash p; \emptyset \uparrow}\,\downarrow\uparrow \qquad \dfrac{\dfrac{p \in \{p\}}{p \vdash p \downarrow}\,Asm}{p \vdash p; \emptyset \uparrow}\,\downarrow\uparrow}{\dfrac{p \vdash p \wedge p; \emptyset \uparrow}{\vdash p \to (p \wedge p); \emptyset \uparrow}\,\to I}\,\wedge I$$

corresponds to the log

```
(impl(p,and(p,p)),impl-i,[],[
    (and(p,p),and-i,[p],[
        (p,tdbu,[p],[
            (p,asm,[p],[])
        ]),
        (p,tdbu,[p],[
            (p,asm,[p],[])
        ])
    ])
]).
```

By `tdbu`, we refer to the top-down/bottom-up conversion rule $\downarrow\uparrow$.

6.2.2 Efficient Backtracking

When traversing the vast search space presented by the many rules available in a natural deduction system, we must expect to frequently encounter dead ends—paths along which the derivation cannot be completed. In such an event, the program will backtrack to the last point at which there are still different untried decisions available and try one of them, and thanks to the loop-free nature of the underlying systems, this process will eventually lead to a complete exploration of the search space. The time needed to do so can be reduced significantly if we avoid redoing those decisions of which we know that changing them will not make any difference.

In $\mathbf{N}^*$, this is quite simple: Once we have a certain sequent $\Gamma \vdash F; \Delta \uparrow$ or $\Gamma \vdash F \downarrow$, how exactly that sequent was derived has no bearing on subsequent inference steps. This means there is no need to bother with backtracking into a derivation that has already succeeded once, which we can ensure by wrapping each derivation step with the predicate `once`.

Unfortunately, $\overline{\mathbf{N}}^*_{BOX}$ is somewhat less accommodating, as the "how" of a top-down derivation can very much make a difference whenever we reach a rule that branches into two of them. For instance, the inference step

$$\dfrac{\Gamma \dashv A; \Delta; \Theta_1 \uparrow \quad A, \Gamma \dashv B; \Delta; \Theta_2 \uparrow \quad A \notin \Gamma \quad BrCon(\Theta_1, \Theta_2, |\Gamma|)}{\Gamma \dashv A \vee B; \Delta; \Theta_1 \cup \Theta_2}\,\vee I_1$$

requires answering a grand total of four questions:

1. Is $A \notin \Gamma$?
2. Is there a derivation of $\Gamma \dashv A; \Delta; \Theta_1 \uparrow$?
3. Is there a derivation of $A, \Gamma \dashv B; \Delta; \Theta_2 \uparrow$?
4. Does $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ hold?

In bottom-up proof search, we can always expect to have Γ fully instantiated before any of the premisses are derived, but the Θ component is filled in the reverse direction and so remains completely unknown until we have completed the derivation in which it appears. Not checking the condition on Γ upfront would only lead to pointlessly wasting time trying to derive the premisses and $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ must be checked at the end since Prolog does not natively provide a way to preemptively impose the required constraint on the unbound variables, so the only flexibility we have with this order would be swapping around Steps 2 and 3.

The conundrum now is as follows: If Step 3 fails, there is no need to backtrack into Step 2, since at this stage the reason for the failure is still unrelated to anything that happened in the derivation of the other premiss. But if Step 4 fails, a different derivation of either premiss could have a different Θ component that resolves the issue, and so backtracking into both steps 2 and 3 becomes necessary for completeness. In other words, we would have to eliminate the choice points left by Step 2 after a failure of Step 3, but retain them if Step 3 succeeds.

We can, in fact, motivate Prolog to process our program in just this way by using a specific combination of its built-in control structures:

```
1,
call((
    2,
    (3 *-> true ; !, false)
)),
4
```

The numbers 1-4 here act as stand-ins for the goals checking the respective condition. The most important line is `(3 *-> true ; !, false)`, which executes Step 3 (the search for a derivation of $A, \Gamma \dashv B; \Delta; \Theta_2 \uparrow$) in the condition part of the `*->` predicate, which unlike the plain `->` predicate (not to be confused with the logical connective $\rightarrow$) leaves all choice points created on its left-hand side intact. Thus, if Step 3 succeeds, we move to Step 4 while keeping the choice points from Step 2 and 3, and will backtrack to them if we find $BrCon(\Theta_1, \Theta_2, |\Gamma|)$ does not hold. However, if Step 3 fails, we instead execute `!, false`, which will first use a cut (`!`) to prune all choice points created by

Steps 2 and 3 (but no others, as `call` is opaque to the cut) and then trigger a failure, backtracking to whatever previous choice points are available.

It is worth noting that choice points from Step 2 and 3 will also remain after Step 4 has succeeded, meaning a failure in a subsequent Step could cause the program to attempt searching for alternate derivations of $\Gamma \dashv A; \Delta; \Theta_1 \uparrow$ or $A, \Gamma \dashv B; \Delta; \Theta_2 \uparrow$. This may actually be useful: The conclusion of $\vee I_1$, for example, has $\Theta_1 \cup \Theta_2$, so changing the Θ component of one of its premisses may resolve a conflict in a rule using that conclusion itself among its premisses.

While necessary for completeness when implementing $\overline{\mathbf{N}}^*_{BOX}$, enabling such extensive backtracking into alternate derivation paths can have disastrous consequences with regards to the quick termination we would ideally like to see. While loop-freeness guarantees us all derivations are finite in length, and thus there is only a finite number of derivations possible for each formula, that number is still potentially enormous. And if those alternatives do no actually differ in the Θ component, the program will waste inordinate amounts of time exploring them only to fail at the consistency check with every single one.

To counteract this issue, we would have to predict in advance which alternate derivations cannot impact Θ and avoid leaving choice points for them in the first place. One simple measure is disregarding alternatives for a derivation by At or $\neg At$ (by placing a cut at the end of the respective clauses). In a simplified normal derivation, the only other way to obtain p or $\neg p$ is from an elimination rule (which we can simply check before the atomic rules) or by CRV or $\neg I$. However, those rules need to find a contradiction either by deriving their intended conclusion again under the opposite assumption, or by already having a contradiction present without those assumptions. In the former case, we would still need the atomic rule and thus not change Θ, and in the latter we could just directly use the contradiction to begin with. So any alternative to such a derivation would be useless to our backtracking efforts, and we can safely discard the choice point right away. Similarly, any part of the derivation that ends on $\Theta = \emptyset$ can be kept intact, as no variation of it could possibly impact the consistency check of subsequent rules any less. We can accomplish this behavior by simply adding `(Ats = [] -> ! ; true)` at the end of each clause.

6.2.3 Top-Down Closures

For each instance of the $\downarrow\uparrow$ rule, we need to find a derivation consisting solely of top-down rules that ends on $\Gamma \vdash F \downarrow$ or $\Gamma \dashv F \downarrow$, with Γ and F matching the respective conclusion. Top-down rules, obviously, are not well-suited to being processed bottom-up starting from the known F, so we must instead take each assumption in Γ, repeatedly apply top-down rules, and hope to eventually find F. In the worst case, we would construct all possible top-down derivations, only to then discard all but the one actually ending on the correct F.

To make this process a little less wasteful, we can instead do the work only once in advance for each assumption scope, storing all found derivations in a *top-down closure* from which $\downarrow\uparrow$ can simply pick without needing to conduct a proof search of its own. Formally, we can define the top-down closure $\texttt{TDC}(\Gamma)$ for $\mathbf{N}^*$ as the smallest set of derivations such that:

1. If $F \in \Gamma$, then $\dfrac{}{\Gamma \vdash F \downarrow}\ \mathit{Asm} \in \texttt{TDC}(\Gamma)$.

2. Let R be a $\mathbf{N}^*$ rule of the form

$$\frac{\Gamma \vdash F_1 \downarrow \quad \cdots \quad \Gamma \vdash F_n \downarrow}{\Gamma \vdash F \downarrow}\ R,$$

and $\delta_1, \ldots, \delta_n$ derivations of $\Gamma \vdash F_1 \downarrow, \ldots, \Gamma \vdash F_n \downarrow$ $(n \geq 1)$. If $\{\delta_1, \ldots, \delta_n\} \subseteq \texttt{TDC}(\Gamma)$, then also

$$\frac{\delta_1 \quad \cdots \quad \delta_n}{\Gamma \vdash F \downarrow}\ R \in \texttt{TDC}(\Gamma).$$

As already established, we do not actually care how exactly a sequent was derived, so in practice this set can be further restricted by omitting duplicate derivations of the same formula F. For the same reason, it makes sense to block backtracking into the computation of the top-down closure by using `once`.

If we swap $\mathbf{N}^*$ for $\overline{\mathbf{N}}^*_{BOX}$ and $\vdash$ for $\dashv$ in the above definition, it also works for the refutational system. Even there, we do not need more than one derivation with the same conclusion, because atomic rules do not factor into top-down reasoning.

Another advantage of always keeping the top-down derivable formulas in view is that we can sometimes avoid redundant steps, which naturally leads to more readable derivations. Specifically, we can replace side conditions of the form $F \notin \Gamma$ that prevent duplicate assumptions with $F \notin \texttt{TDC}(\Gamma)$, since a formula directly derivable from an assumption this way can be used in any place where we would use an assumption. This is especially productive when applying $\vee E$ of $\mathbf{N}^*$ and $\wedge E$ of $\overline{\mathbf{N}}^*_{BOX}$, which can be reduced to just one of their subdeductions. On the other hand, there is no meaning to this trick with rules like $\rightarrow I$ of $\mathbf{N}^*$, which also have a counterpart that does not introduce new assumptions, and so we retain the original side condition for these rules.

6.2.4 Heuristic Rule Selection

Ideally, we do not want to backtrack at all, and instead just immediately choose the correct rule at every fork in the road. This is clearly not realistic without the benefit of hindsight, but perhaps some simple heuristics to guide our proof search will at least improve the accuracy of rule selection.

Intuition suggests that, if multiple rules are applicable to a bottom-up sequent, we should try to use the more specific ones first. For example, in $\mathbf{N}^*$, given $\Gamma \vdash A \wedge B; \Delta \uparrow$, we

would try deriving $\Gamma \vdash A; \Delta \uparrow$ and $\Gamma \vdash B; \Delta \uparrow$ for $\wedge I$ before $\neg(A \wedge B), \Gamma \vdash \bot; \emptyset \uparrow$ for *CRF*, because the former rule is only applicable to conjunctions while the latter can produce any formula. Indeed, using *CRF* first would present the possibility of next trying to derive $\neg(A \wedge B), \Gamma \vdash A \wedge B; \Delta \uparrow$ for $\neg E$, and if that inference is accomplished by $\wedge I$, the *CRF* step inbetween would turn out to have been a needless detour. On the other hand, leading with $\wedge I$ immediately focuses the search on subformulas A and B, and since no bottom-up rules demand formulas of higher degree than their conclusion as premisses (with the exception of $\neg E$ in specific cases), $A \wedge B$ is effectively off the table in these branches. Thus a preference for more specific rules in bottom-up reasoning can be expected to help in arriving at the most straightforward and natural derivations.

However, this approach alone does not actually make it more likely the first rule selected will lead to a succesful derivation without backtracking. Doing so would require us to predict which premisses are actually provable, which to some extent is possible by exploiting the top-down closure that accompanies proof search. Matching the current sequent to prove with the conclusion of some top-down derivation included in the closure means immediately ending the branch successfully, so obviously $\downarrow\uparrow$ should always be tried before all other rules. We can go one step further by prioritizing rules such as the $\mathbf{N}^*$ $\vee I_1$ and $\vee I_2$, which require different premisses for the same conclusion and are equally specific, based on whether or not a derivation of their respective premiss is included in the top-down closure.

A bit of a special case is the *BOX* rule in $\overline{\mathbf{N}}_{BOX}$. While its applicability to any formula whatsoever would lead us to consider it with low priority according to our reasoning so far, the more practical approach is actually to use *BOX* fairly early in the refutational proof search, at least before the potentially backtracking-intensive $\vee I$ and $\rightarrow I$ rules, so that all further derivation steps take place within the scope of the assumption $\bot$. This is because some formulas are not derivable without using *BOX* and enabling full access to atomic rules, while all formulas derivable without this step are also derivable with it. Therefore, using *BOX* by default, despite seemingly introducing a redundant inference, saves us from wasting time on all other rules first in those cases where it actually is required.

6.3 Evaluation

We evaluate not only the performance of Hermes on selected propositional problems, but also how much its derivations for individual formulas resemble those that have been previously produced by humans.

6.3.1 Performance

While our system has no particular focus on speed, and relies far too heavily on extensive backtracking to excel in that area, we still do want to ascertain its ability to produce derivations of various formulas in “reasonable” time. Measurements of performance

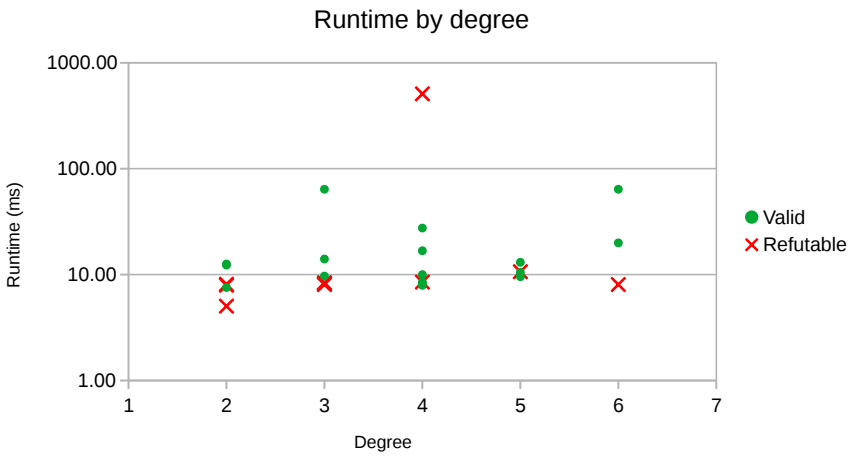


Figure 6.2: Runtime by degree.

also provide a useful quantitative metric by which we can compare the assertional and refutational modules, which ties into our initial aim of seeing how well the strategies known from assertional provers translate to a refutational system.

To this end, we have selected 17 valid and 13 refutable formulas with varying degrees of complexity as input for the experimental evaluation of Hermes, listed in Table 6.1. The valid ones are largely taken from Pelletier's *Seventy-Five Problems for Testing Automatic Theorem Provers* [25], which features several propositional problems built specifically with natural provers in mind. The refutable ones include our running example from Tamminga's original work and some corruptions of valid formulas. Randomized formulas from an online problem collection[1] were also used in both categories. Table 6.2 shows the results—both runtime of the program and length of the resulting derivation—for valid formulas, and table 6.3 for refutable ones. All experiments were carried out by calling the program through SWI-Prolog version 8.4.2 (threaded, 64 bits) on the same machine (Dualcore Intel Pentium 2020M 2.4GHz with 4GB RAM and 2MB CPU cache). Runtime was measured by wrapping the initial calls to `ndprovable` and `ndrefutable` with Prolog's `call_time` predicate, averaged over 3 runs, and capped at 15 seconds. Formulas for which no derivation was found in this time are marked as "Unsolved".

As visualized in Figures 6.2 and 6.3, the runtime of our test cases is more connected to the length of the output derivation than to the degree of the input formula. This result can be considered reflective of the great variety in inference rules and reasoning patterns employed by natural deduction systems, making it so that some large formulas only need to be processed partially while some small formulas require surprisingly roundabout arguments. Derivation length more directly indicates the amount of work actually done,

[1] `https://github.com/ferram/jtabwb_problems/`.

Table 6.1: Formulas used for evaluation.

	Formula	Degree	Status
F_1	$p \vee \neg p$	2	Valid
F_2	$p \wedge \neg p$	2	Refutable
F_3	$(p \to p) \wedge (p \to p)$	2	Valid
F_4	$(p \to q) \vee (q \to p)$	2	Valid
F_5	$(p \to q) \to (q \to p)$	2	Refutable
F_6	$(p \wedge q) \vee p$	2	Refutable
F_7	$((p \to q) \to p) \to p$	3	Valid
F_8	$((p \to q) \to p) \to q$	3	Refutable
F_9	$\neg(p \to q) \to (q \to p)$	3	Valid
F_{10}	$((p \vee q) \to (p \vee r)) \to (p \vee (q \to r))$	3	Valid
F_{11}	$((p \wedge q) \to (p \wedge r)) \to (p \wedge (q \to r))$	3	Refutable
F_{12}	$(\neg p \to p) \wedge (p \to \neg p)$	3	Refutable
F_{13}	$((p \to q) \wedge \neg p) \to \neg q$	3	Refutable
F_{14}	$q \to (((r \vee p) \vee r) \to (p \vee (r \wedge q)))$	4	Valid
F_{15}	$((p \to q) \to (\neg q \to \neg p)) \wedge ((\neg q \to \neg p) \to (p \to q))$	4	Valid
F_{16}	$(\neg\neg p \to p) \wedge (p \to \neg\neg p)$	4	Valid
F_{17}	$(((p \to q) \to (\neg p \vee q)) \wedge ((\neg p \vee q) \to (p \to q))$	4	Valid
F_{18}	$((\neg p \to q) \to (\neg q \to p)) \wedge ((\neg q \to p) \to (\neg p \to q))$	4	Valid
F_{19}	$(F'_{19} \to F''_{19}) \wedge (F''_{19} \to F'_{19})$ $F'_{19} = (p \vee (q \wedge r))$ $F'_{19} = ((p \vee q) \wedge (p \vee r))$	4	Valid
F_{20}	$(((p \to q) \wedge \neg p) \to \neg q) \to (\neg q \to ((p \to q) \wedge \neg p))$	4	Refutable
F_{21}	$(((\neg p \to q) \wedge (\neg p \vee q)) \to ((p \vee \neg q) \to (p \to \neg q))$	4	Refutable
F_{22}	$(((p \vee q) \wedge (\neg p \vee q)) \wedge (p \vee \neg q)) \to \neg(\neg p \vee \neg q)$	5	Valid
F_{23}	$(((q \to r) \wedge (r \to (p \wedge q))) \wedge (p \to (q \vee r))) \to$ $((p \to q) \wedge (q \to p))$	5	Valid
F_{24}	$(F'_{24} \to F''_{24}) \wedge (F''_{24} \to F'_{24})$ $F'_{24} = ((p \to q) \wedge (q \to p))$ $F''_{24} = ((q \vee \neg p) \wedge (\neg q \vee p))$	5	Valid
F_{25}	$(((p \to p) \wedge (p \to p)) \vee ((\neg p \to p) \wedge (p \to \neg p))) \to p$	5	Refutable
F_{26}	$(F'_{26} \to F''_{26}) \wedge (F''_{26} \to F'_{26})$ $F'_{26} = (p \wedge (q \to r)) \to s$ $F''_{26} = ((\neg p \vee q) \vee s) \wedge ((\neg p \vee \neg r) \vee s)$	6	Valid
F_{27}	$(F'_{27} \to F''_{27}) \wedge (F''_{27} \to F'_{27})$ $F'_{27} = ((((p \to q) \wedge (q \to p)) \to r) \wedge (r \to ((p \to q) \wedge (q \to p))))$ $F''_{27} = ((p \to ((q \to r) \wedge (r \to q))) \wedge (((q \to r) \wedge (r \to q)) \to p))$	6	Valid
F_{28}	$((\neg(p \wedge \neg p) \to p) \wedge (p \to \neg(p \wedge \neg p))) \wedge \neg p$	6	Refutable
F_{29}	$(p \to F'_{29}) \wedge (F'_{29} \to p)$ $F'_{29} = (((p \to p) \wedge (p \to p)) \vee ((\neg p \to p) \wedge (p \to \neg p)))$	6	Refutable
F_{30}	$(F'_{29} \to p) \wedge (p \to F'_{29})$	6	Refutable

Table 6.2: Test results on valid formulas.

Formula	Runtime	Lines
F_1	7.54 ms	11
F_3	12.29 ms	7
F_4	12.58 ms	20
F_7	9.67 ms	22
F_9	14.02 ms	11
F_{10}	63.91 ms	44
F_{14}	27.49 ms	36
F_{15}	9.98 ms	23
F_{16}	7.91 ms	15
F_{17}	8.48 ms	22
F_{18}	8.35 ms	23
F_{19}	16.78 ms	74
F_{22}	9.54 ms	47
F_{23}	10.47 ms	44
F_{24}	13.04 ms	63
F_{26}	19.90 ms	225
F_{27}	63.93 ms	181

Table 6.3: Test results on refutable formulas.

Formula	Runtime	Lines
F_2	5.03 ms	2
F_5	8.10 ms	14
F_6	7.93 ms	7
F_8	8.22 ms	14
F_{11}	8.20 ms	13
F_{12}	8.04 ms	11
F_{13}	8.37 ms	18
F_{20}	8.49 ms	26
F_{21}	508.19 ms	38
F_{25}	10.63 ms	22
F_{28}	8.05 ms	13
F_{29}	Unsolved	
F_{30}	8.31 ms	23

but even this metric does not capture the significant effort of backtracking from failed reasoning paths.

This hidden backtracking effort is also the explanation for the most striking outlier, the refutable formula

$$F_{21} = (((\neg p \to q) \wedge (\neg p \vee q)) \to ((p \vee \neg q) \to (p \to \neg q))).$$

It takes the program over half a second to produce a derivation showing the refutability of this formula, because the initial Θ_1 and Θ_2 for the premisses of its effective root rule $\to I_1$ (disregarding BOX) are in conflict. Therefore, we try to find an alternative derivation of the second premiss $((p \vee \neg q) \to (p \to \neg q))$ to alter Θ_2, and since this formula itself has $\to$ as its outermost connective, further branching is possible and compounds the issue. In the case of F_{21}, we still succeed in finding a derivation whose atomic rule applications do not conflict with those of the other branch within somewhat reasonable time, but one only needs to look at the unsolved F_{29} to see this is not guaranteed.

In fact, the two refutable formulas

$$F_{29} = (p \to F'_{29}) \wedge (F'_{29} \to p),$$
$$F_{30} = (F'_{29} \to p) \wedge (p \to F'_{29})$$

demonstrate quite clearly how backtracking from failed branches is the major issue hindering the performance of the refutational system. While proof search for F_{29} fails to

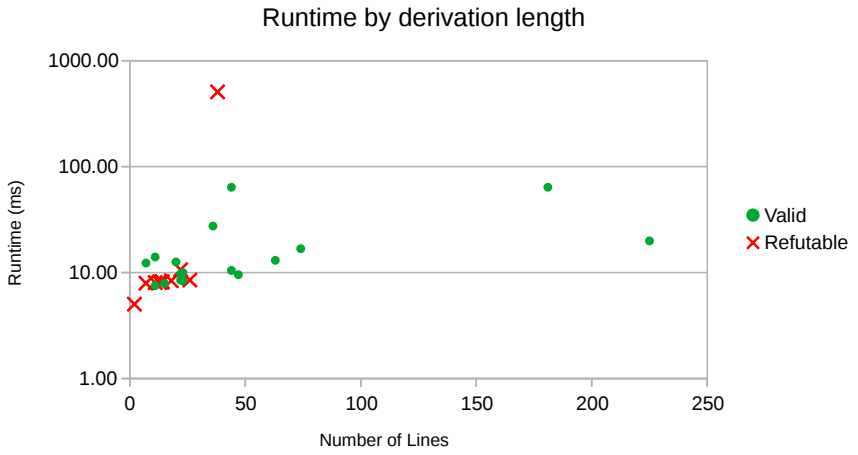


Figure 6.3: Runtime by derivation length.

complete within the allotted 15 seconds, merely switching the conjuncts to produce F_{30} gives us a result in milliseconds. This is simply because $\wedge I_1$ is the first $\wedge$ introduction rule used by default, and $(p \to F'_{29})$ happens to be a valid formula, so we must await the failure of all possible derivation paths for it—including the countless variations that only alter a Θ component—before finally considering the easily refutable $(F'_{29} \to p)$.

More generally, we can observe that derivations of refutability tend to be found slightly faster than their assertional counterparts, and are much shorter despite similarly complex input formulas. This can likely be attributed to atomic rules as well, since they provide an immediate way to complete a branch of reasoning once the formula has become reduced to any propositional constant or its negation, without needing to rely on the presence of specific assumptions. At the same time, these rules and the conflicts between them in neighboring branches are the root cause of excessive runtime on certain refutable formulas, making them a truly double-edged sword.

6.3.2 Naturalness

More than optimal performance, our aim in using natural deduction as the basis of an ATP is to have it output derivations that employ human-like—and thus human-readable—forms of reasoning. One easy way to judge this quality is comparing a derivation made by the program to one independently made by a human for the same formula.

We first consider the assertional formula $((p \to q) \wedge (p \to r)) \to (p \to (q \wedge r))$. Prawitz [29] gives a straightforward derivation that can be rendered in **N** as follows:

1.	$(p \to q) \wedge (p \to r)$	assumption
2.	p	assumption
3.	$p \to q$	$\wedge E_1$: 1
4.	p	*Triv*: 2
5.	q	$\to E$: 3, 4
6.	$p \to r$	$\wedge E_2$: 1
7.	p	*Triv*: 2
8.	r	$\to E$: 6, 7
9.	$q \wedge r$	$\wedge I_1$: 5, 8
10.	$p \to (q \wedge r)$	$\to I$: 2–9
11.	$((p \to q) \wedge (p \to r)) \to (p \to (q \wedge r))$	$\to I$: 1–10

The derivation given by the `ndj` module of Hermes is, after omitting some redundant uses of *Triv* to get the premisses for the $\wedge$ elimination rules, line for line identical to this one.

On the refutational side, we can again resort to the derivation of $((p \to q) \wedge \neg p) \to \neg q$ given by Tamminga [39]:

1.	$\neg q$	$\neg At$
2.	$\neg q$	assumption
3.	p	*At*
4.	$(p \to q) \wedge \neg p$	assumption
5.	$p \to q$	assumption
6.	p	$\to E_2$: 5
7.	$\top$	$\neg E$: 2, 6
8.	$\neg p$	assumption
9.	q	*Triv*: 3
10.	$\top$	$\neg E$: 8, 9
11.	$\top$	$\wedge E$: 4, 5–7, 8–10
12.	$\neg((p \to q) \wedge \neg p)$	$\neg I$: 4–11
13.	$((p \to q) \wedge \neg p) \to \neg q$	$\to I_2$: 1, 2–12

For the same formula, Hermes outputs a derivation that translates to $\overline{\mathbf{N}}_{BOX}$ as follows:

1.	$\bot$	assumption
2.	p	*At*
3.	$\neg q$	$\neg$*At*
4.	$(p \to q) \land \neg p$	assumption
5.	$p \to q$	assumption
6.	$\neg p$	$\to E_1$: 5
7.	p	*Triv*: 2
8.	$\top$	$\neg E$: 6, 7
9.	$\neg p$	assumption
10.	p	*Triv*: 2
11.	$\top$	$\neg E$: 9, 10
12.	$\top$	$\land E$: 4, 5–8, 9–11
13.	$\neg((p \to q) \land \neg p)$	$\neg I$: 4–12
14.	$\neg((p \to q) \land \neg p)$	assumption
15.	$\neg q$	*Triv*: 3
16.	$((p \to q) \land \neg p) \to \neg q$	$\to I_1$: 13, 14–15
17.	$((p \to q) \land \neg p) \to \neg q$	*BOX*: 1–16

In this case, the program fails to find the short manual derivation due to first attempting to use $\to I_1$, as neither side of the implication can be found within the top-down closure directly. Also, the redundant layer of *BOX* that is added by default is, in fact, required to make this derivation possible, since otherwise the atomic rules at line 2 and 3 would not be legal to apply anywhere. These deviations result in a slightly longer and more complex derivation, but notably, the core argument—obtaining $\neg((p \to q) \land \neg p)$ by $\neg I$, $\land E$, and $\neg E$—remains intact, the only difference being the use of $\to E_1$ rather than $\to E_2$ at line 6.

Of course, even if the reasoning is fundamentally the same, additional lines and roundabout ways to obtain premisses usually hinder readability, so it would be beneficial to somehow motivate our system to take the short path as well. One possibility would be to prioritize the side of $\to$ that has the lower degree so that the more complex formula receives the benefit of the additional assumption, but it is questionable if this would be an improvement in general, as we have already seen that the degree of a formula is not necessarily related to the difficulty of deriving it. We could also more specifically always try to use the $\to$ elimination rule that allows deriving the first premiss directly by an atomic rule, but even this has potential to backfire if the resulting entry in Θ_1 then conflicts with something we need to derive the second premiss. Applying this particular heuristic only while there are no assumptions present (and thus before even *BOX*) would be safe and indeed resolve the issue at hand, but overall this is an adaptation that would only be relevant on rare occasions.

CHAPTER 7

Conclusion

In this thesis, we have investigated the use of automated theorem proving for natural deduction and its complementary calculus. In particular, through the joint implementation Hermes, we have shown that the same general strategies can be employed to automate both assertional and refutational proof search with such natural systems.

Of central importance to this is the fact that the weak normalisation theorem holds both for the assertional natural deduction calculus $\mathbf{N}$ and for $\overline{\mathbf{N}}_{BOX}$, a slightly altered version of the refutational natural deduction calculus. By taking advantage of the existence of normal derivations for any valid or refutable formula, respectively, we can apply the well-established technique of splitting our proof search between bottom-up and top-down reasoning fragments, thus restricting the search space significantly. Further tweaking of these two normalisable systems gave us $\mathbf{N}^*$ and $\overline{\mathbf{N}}^*_{BOX}$, which add the useful property of loop-freeness to guarantee eventual termination of a proof search procedure.

A brief evaluation of Hermes demonstrated its ability to judge a given propositional formula either valid or refutable, and to back up this judgment with a derivation that quite closely resembles human patterns of reasoning. However, it also made apparent that the system is far from optimal in terms of performance, particularly when it comes to backtracking from failed paths in refutational proof search. Because $\overline{\mathbf{N}}^*_{BOX}$ derivations have to ensure that atomic rule applications in neighbouring branches do not contradict each other, it becomes necessary to also explore alternate derivations for already succeeded premisses in hopes that one of them avoids such a conflict, which quickly gets out of hand. As a result, the practical applicability of our system is currently limited to fairly simple propositional formulas.

An obvious direction for future research would therefore be to resolve these performance issues and create an ATP that can produce natural deduction derivations for the validity or refutability of even more complex formulas. For example, the excessive backtracking on failing refutational paths could be addressed by finding ways to more strictly limit

the premisses for which we have to consider alternate derivations, or by implementing a custom constraint that can be placed on the second premiss of a branching rule to ensure atomic rules cannot be applied in a way that contradicts the first premiss to begin with.

Another significant step towards making the system suitable for applications such as program verification is expanding it from propositional logic to first-order logic, which is already supported by many existing natural deduction ATPs. However, the refutational aspect becomes problematic here: Since first-order logic is only semi-decidable, a calculus that derives exactly those first-order formulas that are not valid cannot possibly be complete. Therefore, if we wanted to expand the capabilities of our ATP while retaining its theoretical completeness, we would have to limit ourselves to some still decidable fragment of the logic, such as two-variable first-order logic [15]. There is, however, also a possibility that accepting this incompleteness and the existence of formulas for which no conclusion can ever be reached would still result in a practically useful system—after all, the many purely assertional first-order provers already operate under this fundamental limitation.

Bibliography

[1] M. Bogojeski and H. Tompits. On sequent-type rejection calculi for many-valued logics. In M. Urbański, T. Skura, and P. Łupkowski, editors, *Reasoning: Games, Cognition, Logic*, pages 193–207. College Publications, 2020.

[2] L. Dafa. A natural deduction automated theorem proving system. In D. Kapur, editor, *Proceedings of the 11th International Conference on Automated Deduction (CADE-11)*, volume 607 of *Lecture Notes in Computer Science*, pages 668–672. Springer, 1992.

[3] L. Dafa and J. Peifa. Automated natural deduction prover and experiments. In D. Galmiche, editor, *Proceedings of the 6th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX '97)*, volume 1227 of *Lecture Notes in Computer Science*, pages 153–157. Springer, 1997.

[4] R. Dutkiewizc. The method of axiomatic rejection for the intuitionistic propositional logic. *Studia Logica*, 48(4):449–459, 1989.

[5] M. Ferrari and C. Fiorentini. Proof-search in natural deduction calculus for classical propositional logic. In *Proceedings of the 24th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2015)*, volume 9323 of *Lecture Notes in Computer Science*, pages 237–252. Springer, 2015.

[6] M. Ferrari and C. Fiorentini. Goal-oriented proof-search in natural deduction for intuitionistic propositional logic. *Journal of Automated Reasoning*, 62(1):127–167, 2019.

[7] M. Ferrari, C. Fiorentini, G. Fiorino, L. Giordano, V. Gliozzi, A. Pettorossi, and G. L. Pozzato. JTabWb: A Java framework for implementing terminating sequent and tableau calculi. *Fundamenta Informaticae*, 150(1):119–142, 2017.

[8] F. Fitch. *Symbolic Logic: An Introduction*. Ronald Press, 1952.

[9] M. Fitting. *Proof Methods for Modal and Intuitionistic Logics*, volume 169 of *Synthese Library*. Springer, 1983.

[10] M. Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, 2nd edition, 1996.

[11] D. M. Gabbay and N. Olivetti. *Goal-Directed Proof Theory*, volume 21 of *Applied Logic Series*. Springer, 2000.

[12] G. Gentzen. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift*, 39(1):176–210, 1935.

[13] V. Goranko. Refutation systems in modal logic. *Studia Logica*, 53(2):299–324, 1994.

[14] J. Harrison. *Handbook of Practical Logic and Automated Reasoning*. Cambridge University Press, 2009.

[15] L. Henkin. Logical systems containing only a finite number of symbols. Report, Department of Mathematics, University of Montreal, 1967.

[16] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge University Press, 2nd edition, 2004.

[17] S. Jaśkowski. Teoria dedukcji oparta na regu lach za lożeniowych. In *Księga pamiątkowa pierwszego polskiego zjazdu matematycznego 1927*, page 36. Polish Mathematical Society, 1929.

[18] S. Jaśkowski. On the rules of suppositions in formal logic. *Studia Logica*, 1(1):5–32, 1934.

[19] D. Kalish and R. Montague. *Logic: Techniques of Formal Reasoning*. Harcourt Brace Jovanovich, 1964.

[20] J. Łukasiewicz. *Aristotle's Syllogistic From the Standpoint of Modern Formal Logic*. Clarendon Press, 2nd edition, 1957.

[21] M. Maurer. Assertional and refutational natural deduction systems for classical logic. Bachelor's thesis, TU Wien, 2020.

[22] J. Oetsch and H. Tompits. Gentzen-type refutation systems for three-valued logics with an application to disproving strong equivalence. In J.P. Delgrande and W. Faber, editors, *Proceedings of the 11th International Conference on Logic Programming and Nonmonotonic Reasoning* (*LPNMR 2011*), volume 6645 of *Lecture Notes in Computer Science*, pages 254–259. Springer, 2011.

[23] D. Pastre. MUSCADET: An automatic theorem proving system using knowledge and metaknowledge in mathematics. *Artificial Intelligence*, 38(3):257–318, 1989.

[24] D. Pastre. Strong and weak points of the MUSCADET theorem prover: Examples from CASC-JC. *AI Communications*, 15:147–160, 2002.

[25] F. J. Pelletier. Seventy-five problems for testing automatic theorem provers. *Journal of Automated Reasoning*, 2(2):191–216, 1986.

[26] F. J. Pelletier. Automated natural deduction in Thinker. *Studia Logica*, 60:3–43, 1998.

[27] D. A. Plaisted. History and prospects for first-order automated deduction. In A. P. Felty and A. Middeldorp, editors, *Proceedings of the 25th International Conference on Automated Deduction* (*CADE-25*), volume 9195 of *Lecture Notes in Computer Science*, pages 3–28. Springer, 2015.

[28] J. L. Pollock. Interest driven suppositional reasoning. *Journal of Automated Reasoning*, 6(4):419–461, 1990.

[29] D. Prawitz. *Natural Deduction: A Proof-Theoretical Study*. Dover Publications, 1965.

[30] T. Racharak and S. Tojo. On explanation of propositional logic-based argumentation system. In A. P. Rocha, L. Steels, and J. van den Herik, editors, *Proceedings of the 13th International Conference on Agents and Artificial Intelligence* (*ICAART 2021*), volume 2, pages 323–332. SCITEPRESS, 2021.

[31] A. Riazanov and A. Voronkov. Vampire. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction* (*CADE-16*), volume 1632 of *Lecture Notes in Computer Science*, pages 292–296. Springer, 1999.

[32] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[33] J.A. Robinson and A. Voronkov, editors. *Handbook of Automated Reasoning, Volumes I and II*. North Holland, 2001.

[34] W. Samek, G. Montavon, A. Vedaldi, L.K. Hansen, and K.-R. Müller, editors. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700 of *Lecture Notes in Computer Science*. Springer, 2019.

[35] S. Schulz, S. Cruanes, and P. Vukmirović. Faster, higher, stronger: E 2.3. In P. Fontaine, editor, *Proceedings of the 27th International Conference on Automated Deduction* (*CADE-27*), volume 11716 of *Lecture Notes in Computer Science*, pages 495–507. Springer, 2019.

[36] W. Sieg and J. Byrnes. Normal natural deduction proofs (in classical logic). *Studia Logica*, 60(1):67–106, 1998.

[37] G. Stålmarck. Normalization theorems for full first order classical natural deduction. *Journal of Symbolic Logic*, 56(1):129–149, 1991.

[38] R. Statman. *Structural Complexity of Proofs*. Dissertation, Stanford University, 1974.

[39] A.M. Tamminga. Logics of rejection: Two systems of natural deduction. *Logique et Analyse*, 37(146):169–208, 1994.

[40] M. Tiomkin. Proving unprovability. In *Proceedings of the 3rd Annual Symposium on Logic in Computer Science* (*LICS '88*), pages 22–26, 1988.

[41] J. Wielemaker, T. Schrijvers, M. Triska, and T. Lager. SWI-Prolog. *Theory and Practice of Logic Programming*, 12(1-2):67–96, 2012.