Proof Theory and Proof Search of Bi-Intuitionistic and Tense Logic

Linda Postniece

A thesis submitted for the degree of
Doctor of Philosophy at
The Australian National University

November 2010

Typeset in Palatino by TEX and LATEX 2_ε.

Except where otherwise indicated, this thesis is my own original work.

Linda Postniece
21 November 2010

Acknowledgements

Firstly, I would like to thank my supervisors Rajeev Goré and Alwen Tiu, for providing much inspiration and support during the research and writing of this thesis. In particular, I appreciate your timely and constructive feedback on the many drafts I inflicted upon you over the years. Raj, I am also thankful for your mentoring and career advice, which helped me eventually find an optimal path. Thank you also to John Lloyd for valuable advice on various research/career matters, and providing feedback on this thesis.

I would like to thank ANU/CECS, for supporting my studies with a PhD scholarship/topup, and for establishing the VC travel grants scheme, which helped fund my travel to AiML 2008 and WoLLIC 2009. The Wood Bledsoe Travel Award helped fund my travel to IJCAR 2008.

To Tarmo Uustalu, thank you for generously hosting me in Tallinn for a week in 2008, and taking time out of your busy schedule to discuss the details of bi-intuitionistic logic with me. It was a pleasure to meet you.

To my friends Dima Kamenetsky, Florian Widmann, Kate Kiseeva and Rowan Martin-Hughes, I will always cherish the memories of our fun times in Canberra. Your moral support and practical help during 2007/2008 was invaluable - I am very grateful for it.

To my boyfriend Steve, thank you for giving me much love and encouraging me to occasionally look on the illogical side of life :)

Finally, thank you to my parents Ieva and Gunnars, who have supported and encouraged my education from an early age, and continued to look after me when I moved to the other side of the world. To my late grandmother Herta, thank you for your many words of wisdom and inspiration; they continue to guide me every single day.

Abstract

In this thesis, we consider bi-intuitionistic logic and tense logic, as well as the combined bi-intuitionistic tense logic. Each of these logics contains a pair of dual connectives, for example, Rauszer's bi-intuitionistic logic [100] contains intuitionistic implication and dual intuitionistic exclusion. The interaction between these dual connectives makes it non-trivial to develop a cut-free sequent calculus for these logics.

In the first part of this thesis we develop a new extended sequent calculus for bi-intuitionistic logic using a framework of derivations and refutations. This is the first purely syntactic cut-free sequent calculus for bi-intuitionistic logic and thus solves an open problem. Our calculus is sound, semantically complete and allows terminating backward proof search, hence giving rise to a decision procedure for bi-intuitionistic logic.

In the second part of this thesis we consider the broader problem of taming proof search in display calculi [12], using bi-intuitionistic logic and tense logic as case studies. While the generality of display calculi makes it an excellent framework for designing sequent calculi for logics where traditional sequent calculi fail, this generality also leads to a large degree of non-determinism, which is problematic for backward proof-search. We control this non-determinism in two ways:

1. First, we limit the structural connectives used in the calculi and consequently, the number of display postulates. Specifically, we work with nested structures which can be viewed as a tree of traditional Gentzen's sequents, called nested sequents, which have been used previously by Kashima [73] and, independently, by Brünnler and Straßburger [17; 21; 20] and Poggiolesi [97] to present several modal and tense logics.

2. Second, since residuation rules are largely responsible for the difficulty in finding a proof search procedure for display-like calculi, we show how to eliminate these residuation rules using deep inference in nested sequents.

Finally, we study the combined bi-intuitionistic tense logic, which contains the well-known intuitionistic modal logic as a sublogic. We give a nested sequent calculus for bi-intuitionistic tense logic that has cut-elimination, and a derived deep inference nested sequent calculus that is complete with respect to the first calculus and where contraction and residuation rules are admissible. We also show how our calculi can capture Simpson's intuitionistic modal logic [104] and Ewald's intuitionistic tense logic [39].

Contents

List of Figures

Chapter 1

Introduction

1.1 Background and motivation

Classical logic can be seen as a "mathematical model of deductive thought" [38]; it is a formal and rigorous approach to building and analysing deductions. In addition to classical logic, there is a large body of work concerning various non-classical logics that are either restrictions or extensions of classical logic. These specialised logics are often motivated by philosophical or practical considerations; they capture reasoning that cannot be captured using classical logic.

Perhaps the best known alternative to classical logic is intuitionistic logic, which dates back to the early 20th century and was developed during the study of constructive mathematics [112]. Because of its constructive nature, intuitionistic logic forms a rigorous foundation for the type theory of programming languages [93] via the Curry-Howard isomorphism [70] between proofs of intuitionistic formulae and terms in the λ-calculus. The λ-calculus is a Turing-complete model of computation, introduced by Church as part of an investigation into the foundations of mathematics [24]; variants of it have been an inspiration to designers of a number of functional programming languages ([109], [79]).

Modal logic [14] is another non-classical logic; it extends classical logic with modalities, such as possibility and necessity. Variants of modal logic also allow temporal constructs such as "some time in the future" [85]. Modal logic and its variants have numerous application areas, such as knowledge representation and reasoning in artificial intelligence and automated verification of hardware and software [14]. For example, description logics, which are syntactic variants of modal logics, provide a rigorous foundation for the semantic web [66; 6].

In this thesis, we will consider two aspects of logic: proof theory and proof search. While proof theory deals with the analysis of proofs [111], proof search is concerned with algorithmic methods for finding proofs [45] and determining whether a particular logical formula is a theorem. Sequent calculi [46] are one of the logical formalisms that serve both purposes well and as a result are widely used. Sequent calculi consist of a number of rules which allow us to form proofs. A very important rule is the cut rule which effectively encodes the transitivity of theoremhood; it has two premises $\Gamma \Rightarrow A$ and $A \Rightarrow \Delta$, and a conclusion $\Gamma \Rightarrow \Delta$:

$$\frac{\Gamma \Rightarrow A \qquad A \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\,\text{cut}$$

When viewed downwards from premises to conclusion, the cut rule reads: if we can derive A from Γ and Δ from A, then we can derive Δ from Γ; we refer to A as the cut formula.

A fundamental question in proof theory is whether for a given sequent calculus, the cut rule can be eliminated from the set of the rules whilst allowing to prove the same theorems as before. If this is the case, the calculus is said to have cut-elimination. Cut-free sequent calculi are also very important for backward proof search, since the non-deterministic nature of the cut rule complicates backward proof search [45]. More specifically, backward proof search is a method of proof search that starts with the conclusion and attempts to prove it by applying sequent rules backwards. Since the cut formula is not present in the conclusion of the cut rule, the theorem proving procedure needs to guess it before it can apply the cut rule.

While traditional proof search methods focus on attempting to prove theorems, there is also a smaller body of research concerned with finding refutations of non-theorems [81; 50; 94]. Refutation calculi are formalisms that are specifically designed to reason about non-provability as a first-class citizen, rather than conclude non-provability as a result of failed proof search.

In this thesis we will study a number of non-classical logics that we call bi-logics; these logics are extensions of intuitionistic and modal logics with converse operators. For example, bi-intuitionistic logic [100] is an extension of intuitionistic logic with the exclusion connective, which is converse to the intuitionistic implication. Bi-intuitionistic logic has applications in programming language theory - while the traditional Curry-Howard isomorphism relates the implication connective to a function [70], the exclusion connective can be used to describe control mechanisms such as co-routines [28; 29]. Another example of a bi-logic is tense logic, which is the extension of modal logic with converse modal operators [85].

While converse operators give bi-logics more expressive power than their individual sublogics, this often comes at the price of more intricate proof search algorithms. Specifically, it means that the traditional mechanisms used to develop cut-free sequent calculi for the sublogics cannot simply be applied to the bi-logics. For example, while there are many cut free sequent calculi for intuitionistic logic (e.g. [46; 36; 34]), bi-intuitionistic logic has lacked a cut-free sequent calculus for around 30 years [95].

In addition to traditional sequent calculi, there are a number of extended sequent calculi mechanisms that are often used in logics where traditional methods do not suffice. For example, display calculi [12] are an extremely general proof-theoretical framework that can cater for a wide range of logics, including modal logic [116] and bi-intuitionistic logic [51; 53]. The most pleasing property of display calculi is that if the rules of the display calculus enjoy eight easily checked conditions, then the calculus is guaranteed to obey cut-elimination [12]. However, these calculi contain a large degree of non-determinism and as a result are not immediately suitable for proof search [75].

1.2 Summary of contributions

Chapter 2 sets the scene for this thesis and gives an introduction to the proof theory and proof search of non-classical logics. In the rest of the thesis, we will develop new extended sequent calculi formalisms that will allow reasoning in logics such as bi-intuitionistic and tense logic where previous calculi have failed. We will do so using two complementary approaches:

- In Part I (Chapter 3) we develop a framework of proofs and refutations as first class citizens, and give sequent calculi rules that allow to combine proofs and refutations in order to achieve a cut-free sequent calculus for bi-intuitionistic logic where traditional calculi fail. We give the first cut-free sequent calculus for bi-intuitionistic logic that is amenable to proof search. We then show our calculus to be sound and complete with respect to the Kripke semantics of bi-intuitionistic logic. Finally, we give a decision procedure for bi-intuitionistic logic based on our calculus.

- In Part II, we consider the broader problem of proof search in display calculi, and use display-like calculi to develop reasoning techniques for bi-intuitionistic logic (Chapters 4 and 5), tense logic (Chapter 6) and a combination of the two logics (Chapter 7).

  We control the non-determinism of proof search in display logic by using the concept of deep inference in nested sequent calculi. Nested sequent calculi [73; 17; 19; 97] operate on structures that are trees of traditional sequents, and deep inference allows us to apply sequent rules to any formula nested deep inside the nested structure.

  Specifically, we show that for a number of bi-logics, deep inference allows us to simulate residuation rules, which are at the heart of display calculi. Since residuation rules are one of the biggest sources of non-determinism in backward proof search for display calculi, our work is a significant first step towards proof search in general display calculi. Our work is the first which establishes a direct correspondence between proofs in a display-like calculus (with explicit residuation rules) and proofs in a deep-inference calculus (with no explicit residuation rules).

  For each of the bi-logics considered, we give two sequent calculi: the first one is derived from a display calculus and the second one uses deep inference and nested sequents. We give a cut-elimination proof for the first calculus, and then show that the second calculus is syntactically complete with respect to the first calculus. Finally, we give a terminating proof search strategy for the second calculus, which yields a decision procedure for the particular bi-logic.

Chapter 8 surveys related work, and Chapter 9 provides some directions for further work and concludes the thesis. Appendix A gives a short introduction to display calculi and reviews a display calculus for bi-intuitionistic logic. Appendix B contains some additional proofs.

Chapter 2

Background and motivation

In this chapter, we place our proposed work in the context of existing research. In Section 2.1, we set the scene with a brief review of the concepts of syntax, semantics and proof calculi, using classical propositional logic as an example. In Section 2.2, we give a brief synopsis of some non-classical logics that are relevant to our work. We then introduce bi-intuitionistic logic in Section 2.2.4 and tense logic in Section 2.2.5. In Section 2.3, we motivate our work by highlighting the shortcomings of some of the existing approaches to proof search in bi-intuitionistic and tense logics.

2.1 Logic

While logic in the most general sense is the study of reasoning and encompasses topics from philosophy, computer science and mathematics, in this thesis we will focus on symbolic logic, which is a "mathematical model of deductive thought" [38].

From now on, we will use the term "logic" in a more narrow sense to mean a particular formal system or *object logic*. We will study a number of such object logics from three different angles: syntax, semantics and proof calculi. The syntax of each logic defines the formal language we use to write down logical statements; the semantics defines the meaning of those statements, and the proof calculi allow us to reason purely syntactically in this logic and construct logical inferences or proofs of logical statements.

We now introduce the concepts of syntax, semantics and proof calculi in more detail, illustrating them using the well-known classical propositional logic as an object logic.

2.1.1 Syntax

The statements of an object logic are called formulas; they are built from a denumerable set *Atoms* of atomic propositions (atoms for short) using logical connectives. The set of all syntactically correct formulas for an object logic is often specified using a BNF grammar.

For example, the formulas *Fml* of classical propositional logic `CPC` are given by the following grammar, where $p \in Atoms$, $\top$ and $\bot$ are logical constants, and the logical

$$
\begin{aligned}
\bar{V}(p) &= V(p) && (2.1.1)\\
\bar{V}(\top) &= \text{true} && (2.1.2)\\
\bar{V}(\bot) &= \text{false} && (2.1.3)\\
\bar{V}(A \wedge B) &= \begin{cases} \text{true} & \text{if } \bar{V}(A) = \text{true and } \bar{V}(B) = \text{true} \\ \text{false} & \text{otherwise} \end{cases} && (2.1.4)\\
\bar{V}(A \vee B) &= \begin{cases} \text{true} & \text{if } \bar{V}(A) = \text{true or } \bar{V}(B) = \text{true} \\ \text{false} & \text{otherwise} \end{cases} && (2.1.5)\\
\bar{V}(A \to B) &= \begin{cases} \text{true} & \text{if } \bar{V}(A) = \text{false or } \bar{V}(B) = \text{true} \\ \text{false} & \text{otherwise} \end{cases} && (2.1.6)\\
\bar{V}(\neg A) &= \begin{cases} \text{true} & \text{if } \bar{V}(A) = \text{false} \\ \text{false} & \text{otherwise} \end{cases} && (2.1.7)
\end{aligned}
$$

Figure 2.1: Semantics of `CPC`

connectives are $\wedge$ (conjunction), $\vee$ (disjunction), $\to$ (implication) and $\neg$ (negation):

$$A := p \mid \top \mid \bot \mid A \wedge A \mid A \vee A \mid A \to A \mid \neg A.$$

For example, if p_0 is an atomic proposition "the bicycle is blue" and p_1 is an atomic proposition "the sun is shining", then the `CPC` formula $p_0 \to p_1$ formally represents the statement "*if* the bicycle is blue *then* the sun is shining".

We will often need to refer to meta-formulae which define the shape of a formula and can be instantiated to specific formula instances; we call such meta-formulae *formula schemes* and use capital letters to denote them. For example, $A \to (B \to A)$ is a formula scheme and both $p_0 \to (q_0 \to p_0)$ and $(p_0 \wedge q_0) \to ((r_0 \vee s_0) \to (p_0 \wedge q_0))$ are instances of it.

We write $\emptyset$ to mean the empty set. Given a formula A and two sets Δ and Γ of formulae, we write Δ, Γ for $\Delta \cup \Gamma$ and we write Δ, A for $\Delta \cup \{A\}$.

2.1.2 Semantics

While the syntax of an object logic defines the formal language we use to represent statements, semantics defines the meaning of those statements. More specifically, the semantics describes how to assign truth values to formulae, as well as what it means for some formula to be a logical consequence of (a set of) other formula(e). Depending on the object logic, its semantics may be given by a range of mathematical structures, from basic truth tables in `CPC` [90] to algebraic [115], relational [77] and game-theoretical [102] semantics of other logics. We will now review the semantics of `CPC`; later in this chapter we will introduce the more intricate semantics of other object logics such as Kripke semantics for modal logic [77].

In `CPC`, the truth values of atomic propositions are given by a *valuation*, which is a function $V : \mathit{Atoms} \to \{\text{true}, \text{false}\}$: for each atom, the valuation tells us whether this atom is true or false[1]. Then the truth value $\bar{V} : Fml \to \{\text{true}, \text{false}\}$ of compound formulae and logical constants is determined as given in Figure 2.1 [38]. For example, using the atoms p_0 and p_1 from the previous section, if $V(p_0) = \text{true}$ and $V(p_1) = \text{false}$, then $\bar{V}(p_0 \to p_1) = \text{false}$ according to Equation 2.1.6.

We say that a formula A is *valid* in `CPC`, if for every valuation V, we have $\bar{V}(A) = \text{true}$. We say that a formula A is *falsifiable* in `CPC`, if there exists a valuation V such that $\bar{V}(A) = \text{false}$. Given a set Γ of `CPC` formulae and a `CPC` formula A, we say that A is a *logical consequence* of Γ and write $\Gamma \vDash A$ if for any valuation V the following holds:

if (for all $B \in \Gamma$ we have $\bar{V}(B) = \text{true}$) then $\bar{V}(A) = \text{true}$.

Note that if $\Gamma = \emptyset$ then $\vDash A$ iff A is valid.

We may define or formulate an object logic semantically, by specifying the set of all valid formulae in this logic.

A well-known valid formula of `CPC` is $A \vee \neg A$; the fact that this formula is valid is known as the *law of excluded middle*. Another law that holds in `CPC` is the *law of non-contradiction*, that is: $A \wedge \neg A$ is always false, that is, every valuation makes $A \wedge \neg A$ false. As we shall see later in this chapter, there are other non-classical object logics where these laws do not hold.

2.1.3 Calculi

We use *proof calculi* for formally expressing inferences in an object logic. Proof theory is an area of logic that deals with the meta-theoretic analysis of such proofs, as well as translations from one formal theory into another [111]. A proof in some object logic can be formally expressed using one of three basic types of logical formalisms: natural deduction, Hilbert calculi and sequent calculi [111]; we often use the terms "derivation" or "deduction" for such proofs. In this thesis, we will mostly focus on sequent calculi and occasionally refer to Hilbert calculi.

2.1.3.1 Hilbert calculi

A Hilbert calculus for an object logic consists of a set of axioms, which are formula schemes, and (a small number of) inference rules. For example, the axioms of a Hilbert calculus for `CPC` [111] are given in Figure 2.2. The only rule of inference in the Hilbert calculus for `CPC` is called *modus ponens* (MP) and allows to infer B from $A \to B$ and A.

A deduction of a formula A in a Hilbert calculus from a set of assumptions (formulae) Γ is often presented as a list of formulas, where each formula is either a member of Γ, or an instance of an axiom, or is obtained from previous members of the list using an inference rule. We write $\Gamma \vdash A$ to mean that A is deducible from Γ. For example, the following is a Hilbert calculus deduction of the formula q_0 from the as-

[1]The `CPC` valuation is often referred to as a "truth assignment" in introductory logic textbooks [90; 38]. However, as we shall see later, the semantics for modal logic and other non-classical logics uses the concept of "valuation" that serves a very similar purpose to the "truth assignment" in `CPC`. Therefore, for consistency reasons, we are using the term "valuation" for all object logics considered in this thesis, including `CPC`.

$$A \to (B \to A) \tag{2.1.8}$$
$$(A \to (B \to C)) \to ((A \to B) \to (A \to C)) \tag{2.1.9}$$
$$A \to A \lor B \tag{2.1.10}$$
$$B \to A \lor B \tag{2.1.11}$$
$$(A \to C) \to ((B \to C) \to (A \lor B \to C)) \tag{2.1.12}$$
$$(A \land B) \to A \tag{2.1.13}$$
$$(A \land B) \to B \tag{2.1.14}$$
$$A \to (B \to (A \land B)) \tag{2.1.15}$$
$$\bot \to A \tag{2.1.16}$$
$$\neg\neg A \to A \tag{2.1.17}$$

Figure 2.2: Hilbert axioms for `CPC`

sumption $p_0 \land (p_0 \to q_0)$ in `CPC` where the right hand column gives justification for each deduction step:

(0)	$p_0 \land (p_0 \to q_0)$	assumption
(1)	$(p_0 \land (p_0 \to q_0)) \to (p_0 \to q_0)$	instance of axiom $(A \land B) \to B$
(2)	$p_0 \to q_0$	apply MP to (1) and (0)
(3)	$(p_0 \land (p_0 \to q_0)) \to p_0$	instance of axiom $(A \land B) \to A$
(4)	p_0	apply MP to (3) and (0)
(5)	q_0	apply MP to (2) and (4)

Hilbert calculi are often used to define or formulate an object logic syntactically: we say that the *theorems* of some object logic are exactly those formulae that can be proven in some Hilbert calculus from the empty set of assumptions.

2.1.3.2 Sequent calculi

A *sequent* is the basic building block of a sequent calculus derivation, and consists of a pair of multi-sets of formulas separated by a "turnstile". For example, $\Gamma \Rightarrow \Delta$ is a *sequent* in Gentzen's sequent calculus for `CPC` [46]; the sequent consists of an antecedent Γ and a succedent Δ. The intuitive reading of a sequent $\Gamma \Rightarrow \Delta$ is that we can deduce some formula in Δ from the set of formulae Γ. That is, the *formula translation* of the sequent $\Gamma \Rightarrow \Delta$ is the `CPC`-formula $\hat{\Gamma} \to \check{\Delta}$: a conjunction of all the members of Γ implies the disjunction of all the members of Δ.

A *rule of inference* in sequent calculus contains one or more premises, which are sequents, and a single conclusion sequent. For example, the following is Gentzen's original rule for implication on the left [46], where $\Gamma \Rightarrow \Delta, A$ and $B, \Gamma \Rightarrow \Delta$ are the premises, and $A \to B, \Gamma \Rightarrow \Delta$ is the conclusion:

$$\frac{\Gamma \Rightarrow \Delta, A \qquad B, \Gamma \Rightarrow \Delta}{A \to B, \Gamma \Rightarrow \Delta} \to_L$$

The reading of the $\to_L$ rule is: if we can deduce Δ, A from Γ and we can deduce Δ from B, Γ then we can deduce Δ from $A \to B, \Gamma$. The formula translation of the $\to_L$ rule is: if $\hat{\Gamma} \to (\check{\Delta} \vee A)$ and $(B \wedge \hat{\Gamma}) \to \check{\Delta}$ then $((A \to B) \wedge \hat{\Gamma}) \to \check{\Delta}$.

Another example of a sequent calculus rule is the rule for conjunction on the left [46], where $\Gamma, A_i \Rightarrow \Delta$ is the premise and $\Gamma, A_1 \wedge A_2 \Rightarrow \Delta$ is the conclusion.

$$\frac{\Gamma, A_i \Rightarrow \Delta}{\Gamma, A_1 \wedge A_2 \Rightarrow \Delta} \wedge_L \quad \text{for } i \in \{1, 2\}$$

Note that this rule has two versions:

$$\frac{\Gamma, A_1 \Rightarrow \Delta}{\Gamma, A_1 \wedge A_2 \Rightarrow \Delta} \wedge_{L1} \qquad\qquad \frac{\Gamma, A_2 \Rightarrow \Delta}{\Gamma, A_1 \wedge A_2 \Rightarrow \Delta} \wedge_{L2}$$

The reading of the $\wedge_L$ rule is: if we can deduce Δ from Γ, A_i for some $i \in \{1, 2\}$, then we can deduce Δ from $\Gamma, A_1 \wedge A_2$.

A rule of inference is specified using formula schemes; the formula that is introduced in the conclusion is called the *principal formula*, the other formulae are called *side formulae*. An instance of a rule contains an instance of the principal formula and instances of every side formula. For example, the following is an instance of the $\to_L$ rule above, where $p_0 \to q_0$ is the principal formula:

$$\frac{r_0 \Rightarrow s_0, t_0, p_0 \qquad q_0, r_0 \Rightarrow s_0, t_0}{p_0 \to q_0, r_0 \Rightarrow s_0, t_0} \to_L$$

Derivations in sequent calculi are trees that consist of sequents, connected using rules of inference. The leaves of the tree are axiomatic sequents of the form $A \Rightarrow A$, and every sequent in the tree is the conclusion from its predecessor(s), and the root of the tree contains the conclusion of the entire derivation. Formally:

- All instances of axioms of the form $A \Rightarrow A$ are derivations.
- For any instance of a rule ρ, if the premises $\Gamma_1 \Rightarrow \Delta_1$ to $\Gamma_n \Rightarrow \Delta_n$ have derivations Π_1 to Π_n for some $n \geq 1$, then the conclusion $\Gamma \Rightarrow \Delta$ has the following derivation:

$$\frac{\begin{matrix} \Pi_1 \\ \Gamma_1 \Rightarrow \Delta_1 \end{matrix} \qquad \cdots \qquad \begin{matrix} \Pi_n \\ \Gamma_n \Rightarrow \Delta_n \end{matrix}}{\Gamma \Rightarrow \Delta} \rho$$

The $\to_L$ rule above is an example of a *logical rule*, because it operates on the logical connectives in a formula, in this case, the $\to$ connective. Sequent calculi can also contain *structural rules*, which operate on the physical structure of the sequent. Figure 2.3 shows Gentzen's original contraction and weakening rules on the left and on the right, as well as several versions of the cut rule, which we shall discuss in the next section.

For example, the contraction left rule (c_L) replaces two copies the principal formula A in the premise with one copy of A in the conclusion, and involves no logical

$$\frac{A, A, \Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta}\, c_L \qquad \frac{\Gamma \Rightarrow \Delta, A, A}{\Gamma \Rightarrow \Delta, A}\, c_R$$

$$\frac{\Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta}\, w_L \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, A}\, w_R$$

$$\frac{\Gamma \Rightarrow B, \Delta \qquad B, \Gamma' \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\, \text{muliplicative cut}$$

$$\frac{\Gamma \Rightarrow B, \Delta \qquad B, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\, \text{additive cut}$$

$$\frac{\Gamma \Rightarrow B^m, \Delta \qquad B^n, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\, \text{multicut}$$

Figure 2.3: Structural rules in Gentzen's sequent calculus for CPC

connectives explicitly.

The following is an example sequent calculus derivation of the formula q_0 from $p_0 \wedge (p_0 \to q_0)$ in CPC:

$$\dfrac{\dfrac{\dfrac{\dfrac{p_0 \Rightarrow p_0 \qquad q_0 \Rightarrow q_0}{p_0, p_0 \to q_0 \Rightarrow q_0}\, \to_L}{p_0, p_0 \wedge (p_0 \to q_0) \Rightarrow q_0}\, \wedge_L}{p_0 \wedge (p_0 \to q_0), p_0 \wedge (p_0 \to q_0) \Rightarrow q_0}\, \wedge_L}{p_0 \wedge (p_0 \to q_0) \Rightarrow q_0}\, c_L$$

Another example of a structural rule is the weakening rule. For example, the intuitive reading of the w_L rule in Figure 2.3 is that if we can deduce some formula in Δ from the set of formulae Γ, then we can still deduce the same formula from the set $\Gamma \cup \{A\}$.

2.1.3.3 The cut rule

In addition to the logical and structural rules, a sequent calculus may contain a cut rule, which effectively encodes the transitivity of derivability. Consider the multiplicative cut rule of Figure 2.3. Since the *cut formula B* need not be a subformula of either Γ or Δ, this rule violates the *subformula property*, which states that all formulae occurring in the premises of a rule are subformulae of the formulae in the conclusion [45]. Moreover, when read upwards, the multiplicative cut rule partitions the antecedent into Γ and Γ' and the succedent into Δ and Δ'. Another version of the cut rule is the additive version (see Figure 2.3), which still violates the subformula property but does not require partitioning the antecedent and succedent of the conclusion.

One of the main contributions of Gentzen was that he showed that the cut rule can be eliminated from the rules of his sequent calculus LK, that is, any derivation that uses the cut rule can be transformed into a derivation of the same sequent that does not use the cut rule [46]. This process is called *cut-elimination*, the resulting derivation is called *cut-free*, and the calculus is said to have cut-admissibility. In general, we say that a rule is *admissible* in a sequent calculus if we can leave out this rule and still derive the same sequents as before. Formally, a rule ρ is admissible in some calculus C if the following holds: if there exists a derivation of $\Gamma \Rightarrow \Delta$ in C+ρ, then there exists a derivation of $\Gamma \Rightarrow \Delta$ in C.

In some sequent calculi where the contraction rule is not admissible, cut-elimination is complicated for derivations that include an instance of contraction immediately above an instance of cut. Then the multicut rule (see Figure 2.3) may be used, where B^m and B^n stand for m and n copies of the formula B respectively.

If $n = m = 1$, multicut becomes the traditional cut rule. If $n \geq 1$ and/or $m \geq 1$, multicut can be seen as a combination of contraction and cut:

$$\dfrac{\dfrac{\Gamma \Rightarrow B^m, \Delta}{\Gamma \Rightarrow B, \Delta}\, m \times c_R \qquad \dfrac{B^n, \Gamma \Rightarrow \Delta}{B, \Gamma \Rightarrow \Delta}\, n \times c_L}{\Gamma \Rightarrow \Delta}\, cut$$

It is often easier to prove multicut-elimination than cut-elimination [111].

Cut-admissibility has a number of important corollaries. We briefly note some of them here; full details are given by Troelstra and Schwichtenberg [111]. Firstly, cut-admissibility implies the *separation property*: any derivable sequent $\Gamma \Rightarrow \Delta$ always has a derivation using only the logical rules and/or axioms for the logical operators occurring in $\Gamma \Rightarrow \Delta$. This property is helpful if we wish to find a derivation of some sequent $\Gamma \Rightarrow \Delta$, since it limits the number of axioms and rules we need to consider. Secondly, cut-admissibility often implies *Craig's interpolation* [26; 111], which is an important proof-theoretic result on its own and also has applications in software and hardware verification [86]. Formally, Craig's interpolation theorem for `CPC` states: if the sequent $\emptyset \Rightarrow A \rightarrow B$ is derivable then there is a formula I such that the sequents $\emptyset \Rightarrow A \rightarrow I$ and $\emptyset \Rightarrow I \rightarrow B$ are derivable, and every atom of I occurs in both A and B.

An important property of a sequent calculus rule is *invertibility*. Formally, a rule ρ is invertible if the derivability of the conclusion implies the derivability of the premise(s). For example, the contraction rules c_L and c_R in Figure 2.3 are invertible in LK, but the weakening rules w_L or w_R are not.

2.1.3.4 Backward proof search

While proof theory is an important research area on its own, it also has applications in automated deduction, which is concerned with algorithmic methods for proving theorems [45]. Increasingly, sequent calculi are used to decide theoremhood by applying the rules in a backward fashion. Thus it has become important to study such calculi from this proof search perspective. In backward proof search, we start with the sequent we wish to prove, and apply rules backwards to obtain premises, and so on, until we have reduced each premise to an axiomatic leaf sequent. For example, if we wanted to prove the sequent $A \rightarrow B, \Gamma \Rightarrow \Delta$, we would use the $\rightarrow_L$ rule backwards and obtain the premises $\Gamma \Rightarrow \Delta, A$ and $B, \Gamma \Rightarrow \Delta$, which we would then attempt to prove. Notice that the formulae A and B in the premises are strict subformulae of $A \rightarrow B$, meaning that the subgoals $\Gamma \Rightarrow \Delta, A$ and $B, \Gamma \Rightarrow \Delta$ are somehow simpler than the original goal $A \rightarrow B, \Gamma \Rightarrow \Delta$.

However, the structural rules are problematic for backward proof search. For example, if we were to apply the contraction rule backwards with principal formula A to the sequent $A, \Gamma \Rightarrow B$, we would obtain the premise $A, A, \Gamma \Rightarrow B$, and so on, thus potentially causing an infinite loop in the theorem proving procedure. Therefore the use of contraction rules needs to be controlled to avoid this problem. This is done, for example, by Dyckhoff: his LJT [36] is a contraction-free sequent calculus that can be used for proof-search in intuitionistic logic, which we shall introduce later in this chapter. Additionally, once we have shown that the contraction rule is admissible in some sequent calculus, we may consider sequents as pairs of sets of formulae rather than pairs of multisets of formulae. The use of sets instead of multisets can lead to a simpler backward proof search procedure, therefore contraction admissibility is a desirable feature of a sequent calculus.

The weakening rules also pose problems for backward proof search due to their non-deterministic nature: when we attempt to prove a sequent $\Gamma \Rightarrow \Delta$, both the weakening left and weakening rules are applicable to any formula in Γ or Δ. However, the weakening rules are not invertible, meaning that applying them too early might mean losing formulae that are essential for finding the derivation. Fortunately, weakening can often can be shown admissible quite easily, if we build it into the axioms. That is, the axioms of the sequent calculus become $\Gamma, A \Rightarrow A, \Delta$ instead of $A \Rightarrow A$. In terms of backward proof search, this modification means we postpone the weakening step for as long as possible, which allows us to keep all formulae potentially relevant to the derivation we are looking for.

Of all the structural rules, the cut rule is the most problematic from a backward proof search perspective. Since the cut formula B does not appear in the conclusion of the rule, the theorem proving procedure needs to guess it. This is because the cut rule violates the subformula property. Moreover, if the multiplicative version of the cut rule is being used, the theorem proving procedure needs to guess the partitioning of the antecedent into Γ and Γ' and the succedent into Δ and Δ'. As Gallier [45] points out, Gentzen's cut-elimination result [46] is very important for backward proof search, because in cut-free derivations all inferences are purely mechanical and require no ingenuity. Since we are looking for proof search algorithms for a range of object logics that we shall introduce shortly, a large part of our work will consist of developing cut-free sequent calculi, and calculi with cut-elimination, for these logics.

2.1.3.5 Connection to semantics

When developing a proof system such as a sequent calculus for an object logic, we typically want to ensure that this proof system allows us to derive exactly those formulae that are semantically valid in this logic.

More formally, we first want to show that if a formula is syntactically derivable in the calculus, then it is semantically valid - this property of a calculus is called *soundness* and is typically quite easy to show. Secondly, we want to show that if a formula is semantically valid, then it is syntactically derivable in the calculus - this property is called *completeness* and is often non-trivial to show.

We typically also want to link logical consequence with derivability from a set of assumptions. That is, we want to show that we can derive A from the set of assumptions Γ if and only if A is logical consequence of Γ. In this thesis, we will focus only on validity and therefore devise proof calculi that allow us to derive a formula (or a sequent) rather than a formula from a set of assumptions.

2.2 Non-classical logics

In addition to classical propositional logic, there is a large body of work concerning various non-classical logics that are either restrictions or extensions of classical logic: modal, intuitionistic, paraconsistent logic to name a few. These specialised logics are often motivated by philosophical or practical considerations and capture reasoning that cannot be captured by classical logic.

2.2.1 Modal logic

Modal logic extends classical logic with modalities, such as possibility and necessity; variants of modal logic also allow temporal constructs such as "some time in the future". Modal logic and its variants have numerous application areas, for example, knowledge representation and reasoning in artificial intelligence and automated verification of hardware and software [14].

2.2.1.1 Syntax

The basic modal logic $\mathtt{K}$ consists of the connectives of classical propositional logic, as well as modal connectives diamond $\Diamond$ and box $\Box$, where the reading of $\Diamond A$ is "possibly A" and the reading of $\Box A$ is "necessarily A". Formally, formulas of modal logic $\mathtt{K}$ are given by the following grammar, where p is an atom:

$$A := p \mid \top \mid \bot \mid A \land A \mid A \lor A \mid A \to A \mid \neg A \mid \Diamond A \mid \Box A.$$

2.2.1.2 Semantics

The semantics of modal logic $\mathtt{K}$ due to Kripke [76] uses graphs and is referred to as possible world semantics. A Kripke model is a graph consisting of worlds connected by a reachability relation, and each world has a different valuation for propositional atoms. While the truth value of atoms and propositional formulae is determined at a single world, the truth value of modal formulae is determined by examining neighbouring worlds.

More formally, a $\mathtt{K}$-*frame* is a pair $\langle W, R\rangle$, with W a non-empty set (of worlds) and $R \subseteq W \times W$. A $\mathtt{K}$-model is a triple $\langle W, R, V\rangle$, with $\langle W, R\rangle$ a $\mathtt{K}$ frame and $V : \mathit{Atoms} \to$

$w \Vdash \top$		for all w	$w \nVdash \bot$		for all w
$w \Vdash \neg A$	iff	$w \nVdash A$	$w \Vdash A \to B$	iff	$w \nVdash A$ or $w \Vdash B$
$w \Vdash A \vee B$	iff	$w \Vdash A$ or $w \Vdash B$	$w \Vdash A \wedge B$	iff	$w \Vdash A$ and $w \Vdash B$
$w \Vdash \Box A$	iff	$\forall u.$ if wRu then $u \Vdash A$	$w \Vdash \Diamond A$	iff	$\exists u.wRu$ and $u \Vdash A$

Figure 2.4: Forcing of K-formulae

2^W a valuation mapping each atom to the set of worlds where it is true[2]. We use 2^W to mean the powerset of the set W.

For a world $w \in W$ and an atom $p \in Atoms$, if $w \in V(p)$ then we write $w \Vdash p$ and say p is forced by w; otherwise we write $w \nVdash p$ and say p is rejected by w. Forcing of compound formulae is defined by mutual recursion in Figure 2.4. A K-formula A is valid if and only if it is forced by all worlds in all models, i.e. if and only if $w \Vdash A$ for all $\langle W, R, V\rangle$ and for all $w \in W$. A K-formula A is falsifiable if and only if some world in some model rejects A, i.e. if and only if $w \nVdash A$ for some $\langle W, R, V\rangle$ and some $w \in W$.

2.2.1.3 Calculi

The Hilbert axioms for K are all those of CPC plus the following axiom [44]:

$$\Box(A \to B) \to (\Box A \to \Box B) \tag{2.2.1}$$

In addition to the modus ponens inference rule, the Hilbert calculus for K has the *necessitation* rule: infer $\Box A$ from A. Modal logic K is thus a *conservative extension* of CPC, meaning that every formula B of K such that B consists only of CPC connectives is a theorem of K if and only if B is a theorem of CPC.

As Fitting [44] points out, axiomatic proof calculi are elegant, but often difficult for proof discovery. On the other hand, sequent calculi (and their close cousins tableaux calculi [52]) are suitable and therefore widely used for proof search in modal logic. A basic sequent calculus for modal logic consists of all the rules of Gentzen's LK [46] plus the following two rules where $\Box\Gamma = \{\Box B \mid B \in \Gamma\}$ and $\Diamond\Delta = \{\Diamond B \mid B \in \Delta\}$ [44]:

$$\frac{A, \Gamma \Rightarrow \Delta}{\Diamond A, \Box\Gamma \Rightarrow \Diamond\Delta}\,\Diamond_L \qquad \frac{\Gamma \Rightarrow \Delta, A}{\Box\Gamma \Rightarrow \Diamond\Delta, \Box A}\,\Box_R$$

For example, the following is a derivation of axiom 2.2.1:

$$\cfrac{\cfrac{\cfrac{\cfrac{A \Rightarrow A \qquad B \Rightarrow B}{A \to B, A \Rightarrow B}\,\to_L}{\Box(A \to B), \Box A \Rightarrow \Box B}\,\Box_R}{\Box(A \to B) \Rightarrow \Box A \to \Box B}\,\to_R}{\Rightarrow \Box(A \to B) \to (\Box A \to \Box B)}\,\to_R$$

[2]Note the subtle difference between a CPC valuation and a K valuation. Since the CPC models effectively consist of only one world, a CPC valuation maps each atom to $\{\text{true}, \text{false}\}$.

2.2.1.4 Extensions

In addition to the basic modal logic K, there are a variety of extensions of it [44]. These extensions are usually defined either axiomatically (by specifying additional Hilbert axioms to those for K), or semantically as conditions on the frames. For example, the modal logic S4 extends K by adding the following two axioms to the Hilbert calculus for CPC:

$$4: \quad \Box A \to \Box\Box A \tag{2.2.2}$$
$$T: \quad A \to \Diamond A \tag{2.2.3}$$

The frames of S4 are pairs $\langle W, R\rangle$, with W a non-empty set (of worlds) and $R \subseteq W \times W$, such that R is reflexive and transitive.

2.2.2 Intuitionistic logic

Perhaps the most well-known alternative to classical logic is *intuitionistic logic*, which dates back to the early 20th century and was developed during the study of constructive mathematics [112]. Reasoning in *intuitionistic logic* can be seen as a process of constructing mathematical structures [15]: a formula C is a theorem if we can construct a concrete proof of C. For example, we could prove $A \vee \neg A$ if we could construct a proof of the fact that A holds, or if we could construct a proof of the fact that $\neg A$ holds. Thus, $A \vee \neg A$ need not be true in intuitionistic logic, and hence the law of the excluded middle does not hold in intuitionistic logic.

Because of its constructive nature, intuitionistic logic forms a rigorous foundation for the type theory of programming languages [93] via the Curry-Howard isomorphism [70] between proofs of intuitionistic formulae and terms in the λ-calculus. The λ-calculus is a Turing-complete model of computation, introduced by Church as part of an investigation into the foundations of mathematics [24]; variants of it have been an inspiration to designers of a number of functional programming languages ([109], [79]).

2.2.2.1 Syntax

Formally, formulas of intuitionistic logic Int are given by the following grammar, where p is an atom:

$$A := p \mid \top \mid \bot \mid A \wedge A \mid A \vee A \mid A \to A \mid \neg A.$$

2.2.2.2 Semantics

Kripke has also presented possible world semantics for intuitionistic logic [78]; they have many similarities to the semantics of S4. An Int frame is a pair $\langle W, \leq\rangle$, where W is a non-empty set of worlds and $\leq$ is a reflexive and transitive binary accessibility relation. An Int model $M = \langle W, \leq, V\rangle$ is an Int frame $\langle W, \leq\rangle$ together with a valuation

$w \Vdash \top$		for all w
$w \dashV \bot$		for all w
$w \Vdash A \vee B$	iff	$w \Vdash A$ or $w \Vdash B$
$w \dashV A \vee B$	iff	$w \dashV A$ and $w \dashV B$
$w \Vdash A \wedge B$	iff	$w \Vdash A$ and $w \Vdash B$
$w \dashV A \wedge B$	iff	$w \dashV A$ or $w \dashV B$
$w \Vdash \neg A$	iff	$\forall u \geq w \,.\, u \dashV A$
$w \dashV \neg A$	iff	$\exists u \geq w \,.\, u \Vdash A$
$w \Vdash A \to B$	iff	$\forall u \geq w \,.\, u \dashV A$ or $u \Vdash B$
$w \dashV A \to B$	iff	$\exists u \geq w \,.\, u \Vdash A$ and $u \dashV B$

Figure 2.5: Forcing and rejecting of `Int`-formulae

$V : Atoms \to 2^W$ which obeys the *persistence* property:

$$\forall u, w \in W. \forall p \in Atoms \, (\text{if } w \leq u \text{ and } w \in V(p) \text{ then } u \in V(p)) \,.$$

For some statement X and some fixed $w \in W$, we sometimes use the abbreviations:

$\forall u \geq w.X$	means	$\forall u \in W.$ if $w \leq u$ then X
$\forall u \leq w.X$	means	$\forall u \in W.$ if $u \leq w$ then X
$\exists u \geq w.X$	means	$\exists u \in W. w \leq u$ and X
$\exists u \leq w.X$	means	$\exists u \in W. u \leq w$ and X.

Given a model $M = \langle W, \leq, V \rangle$, a world $w \in W$ and an atom $p \in Atoms$, we write $w \Vdash p$ (w forces p) iff $w \in V(p)$, and we write $w \dashV p$ (w rejects[3] p) iff $w \notin V(p)$. We define forcing and rejecting of compound formulae by mutual recursion in Figure 2.5.

2.2.2.3 Calculi

The Hilbert calculus for `Int` contains all the Hilbert axioms of `CPC` (as presented in Figure 2.2), except the law of double negation (axiom 2.1.17). The omission of the law of double negation prohibits us from proving formulae such as $A \vee \neg A$ in `Int`.

A sequent calculus for `Int` may be obtained from Gentzen's LK for `CPC` via either of the following modifications:

1. Restrict all sequents of LK to at most one formula in the succedent; the resulting

[3]Note that we deliberately use $\dashV$ for rejection in intuitionistic, dual intuitionistic and bi-intuitionistic logic, as opposed to $\nVdash$ for rejection in modal logic. We do this because it makes it easier to extend the notions of forcing and rejection to sets of formulae in Definition 2.2.2. While it is the case that a single formula is either forced or rejected by a world, a set of formulae may be forced, rejected or neither forced nor rejected by a world. Therefore we define rejection in its own right, rather than simply as a negation of forcing.

sequent calculus is called LJ [46]. This approach is very common and is often referred to as *singletons on the right*. For example, the following are Gentzen's implication left and right rules in LJ:

$$\frac{\Gamma \Rightarrow A \qquad \Gamma, B \Rightarrow C}{\Gamma, A \to B \Rightarrow C} \to_L \qquad \frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \to B} \to_R$$

2. Replace Gentzen's implication right rule with a rule which has a single formula in the succedent of the premise but allows multiple formulae in the succedent of the conclusion, as shown below:

$$\frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \to B, \Delta} \to_R$$

This approach has been used by Maehara [83] and Dragalin [34] to present multi-succedent sequent calculi for `Int`.

2.2.3 Dual-intuitionistic logic

While intuitionistic logic rejects the law of excluded middle, *paraconsistent logic* is another non-classical logic which rejects the law of non-contradiction. That is, in paraconsistent logic, the formula $A \wedge \sim A$ need not be false; we use $\sim$ to emphasize that paraconsistent negation is different from classical negation. The main feature of paraconsistent logic is that it allows reasoning in the presence of inconsistent information: whilst in classical logic $A \wedge \neg A$ is always false irrespective of A, this is not the case in paraconsistent logic [8]. One example of a paraconsistent logic is dual intuitionistic logic [114; 49; 30].

2.2.3.1 Syntax

Formally, formulas of dual-intuitionistic logic `DualInt` are given by the following grammar, where p is an atom:

$$A := p \mid \top \mid \bot \mid A \wedge A \mid A \vee A \mid A \prec A \mid \sim A.$$

The $\prec$ connective is called "exclusion", and also known as "subtraction" or "co-implication" in the literature. Intuitively, the formula $B \prec C$ reads "B excludes C". We can define paraconsistent negation $\sim A$ using exclusion as $\top \prec A$.

2.2.3.2 Semantics

Several authors have given algebraic semantics of `DualInt` (e.g. [100; 49]). We now present a Kripke semantics that is dual to the Kripke semantics for `Int` and is a subset of Rauszer's semantics for `BiInt` [100], which we shall discuss later in this chapter.

A `DualInt` frame is a pair $\langle W, \leq \rangle$, where W is a non-empty set of worlds and $\leq$ is a reflexive and transitive binary accessibility relation. A `DualInt` model $M = \langle W, \leq, V \rangle$

$w \Vdash \top$		for all
$w \dashv\!\!\mid \bot$		for all w
$w \Vdash A \vee B$	iff	$w \Vdash A$ or $w \Vdash B$
$w \dashv\!\!\mid A \vee B$	iff	$w \dashv\!\!\mid A$ and $w \dashv\!\!\mid B$
$w \Vdash A \wedge B$	iff	$w \Vdash A$ and $w \Vdash B$
$w \dashv\!\!\mid A \wedge B$	iff	$w \dashv\!\!\mid A$ or $w \dashv\!\!\mid B$
$w \Vdash \sim A$	iff	$\exists u \leq w \,.\, u \dashv\!\!\mid A$
$w \dashv\!\!\mid \sim A$	iff	$\forall u \leq w \,.\, u \Vdash A$
$w \Vdash A \prec B$	iff	$\exists u \leq w \,.\, u \Vdash A$ and $u \dashv\!\!\mid B$
$w \dashv\!\!\mid A \prec B$	iff	$\forall u \leq w \,.\, u \dashv\!\!\mid A$ or $u \Vdash B$

Figure 2.6: Forcing and rejecting of `DualInt`-formulae

is an `DualInt` frame $\langle W, \leq \rangle$ together with a valuation $V : Atoms \to 2^W$ which obeys the *reverse persistence* property:

$$\forall u, w \in W. \forall p \in Atoms \, (\text{if } w \leq u \text{ and } u \notin V(p) \text{ then } w \notin V(p)) \, .$$

Given a model $M = \langle W, \leq, V \rangle$, a world $w \in W$ and an atom $p \in Atoms$, we write $w \Vdash p$ (w forces p) iff $w \in V(p)$, and we write $w \dashv\!\!\mid p$ (w rejects p) iff $w \notin V(p)$. We define forcing and rejecting of compound formulae by mutual recursion in Figure 2.6.

Comparing Figures 2.5 and 2.6, we can observe the semantic differences between $\to$ and $\prec$: while the $\to$ connective "looks forward" along the relation $\leq$, the $\prec$ connective "looks backward" along the relation $\leq$.

2.2.3.3 Calculi

In a sequent setting, the introduction rules for exclusion are symmetric to the introduction rules for implication. Using a singletons on the left approach [114; 49; 30], the introduction rules for exclusion are shown below:

$$\frac{A \Rightarrow B, \Delta}{A \prec B \Rightarrow \Delta} \prec_L \qquad \frac{C \Rightarrow A, \Delta \qquad B \Rightarrow \Delta}{C \Rightarrow A \prec B, \Delta} \prec_R$$

We could also develop a Dragalin-like sequent calculus for `DualInt` where the $\prec_L$ rule (as shown below) has a single formula on the left of the premise but allows multiple side formulae on the left of the conclusion:

$$\frac{A \Rightarrow B, \Delta}{\Gamma, A \prec B \Rightarrow \Delta} \prec_L$$

2.2.4 Bi-intuitionistic logic

We now introduce bi-intuitionistic logic, which is one of the two main object logics studied in this thesis.

2.2.4.1 History

Bi-intuitionistic logic (`BiInt`), also known as subtractive logic and Heyting-Brouwer logic, is the union of intuitionistic logic and dual intuitionistic logic; it was introduced by Rauszer as a Hilbert calculus with Kripke semantics [100]. `BiInt` is a conservative extension of both its `Int` and `DualInt` components, and Rauszer's Hilbert calculus also includes interaction axioms between the `Int`-connectives and `DualInt`-connectives. Recently, Pinto and Uustalu have suggested that `BiInt` has potential applications in the area of type theory [95]; Filinski [43] and Crolard [28] have already done initial investigations in this direction. Additionally, Curien and Herbelin [29], and Ariola et al [4] have studied the type theory of classical logic with the exclusion connective.

2.2.4.2 Syntax

Formally, formulas of bi-intuitionistic logic are given by the following grammar, where p is an atom:

$$A := p \mid \top \mid \bot \mid A \wedge A \mid A \vee A \mid A \to A \mid \neg A \mid A \prec A \mid \sim A.$$

The connectives $\to$ and $\neg$ are those of intuitionistic logic, the connectives $\prec$ and $\sim$ are those of dual intuitionistic logic and the connectives $\vee$ and $\wedge$ are from both.

2.2.4.3 Relationship to classical logic

Note that because of conservativity over `Int`, `BiInt` is different from classical logic with an exclusion connective. Similarly, conservativity over `DualInt` means that the exclusion connective is independent and cannot be defined using the other connectives. However, `BiInt` can be seen as being closer to classical logic than either `Int` or `DualInt` is, as evidenced by the following two theorems of `BiInt` which combine the two negations:

$$A \to \sim \neg A \tag{2.2.4}$$

$$\neg \sim A \to A \tag{2.2.5}$$

Moreover, the law of excluded middle holds for the `DualInt`-fragment of `BiInt`: the following formula is a theorem of `BiInt`, if A contains only the `DualInt`-connectives:

$$A \vee \sim A \tag{2.2.6}$$

$$
\begin{array}{lll}
w \Vdash \top & & \text{for all } w \\
w \dashV \bot & & \text{for all } w \\
\\
w \Vdash A \vee B & \text{iff} & w \Vdash A \text{ or } w \Vdash B \\
w \dashV A \vee B & \text{iff} & w \dashV A \text{ and } w \dashV B \\
\\
w \Vdash A \wedge B & \text{iff} & w \Vdash A \text{ and } w \Vdash B \\
w \dashV A \wedge B & \text{iff} & w \dashV A \text{ or } w \dashV B \\
\\
w \Vdash \neg A & \text{iff} & \forall u \geq w \,.\, u \dashV A \\
w \dashV \neg A & \text{iff} & \exists u \geq w \,.\, u \Vdash A \\
\\
w \Vdash A \rightarrow B & \text{iff} & \forall u \geq w \,.\, u \dashV A \text{ or } u \Vdash B \\
w \dashV A \rightarrow B & \text{iff} & \exists u \geq w \,.\, u \Vdash A \text{ and } u \dashV B \\
\\
w \Vdash \sim A & \text{iff} & \exists u \leq w \,.\, u \dashV A \\
w \dashV \ \sim A & \text{iff} & \forall u \leq w \,.\, u \Vdash A \\
\\
w \Vdash A \mathrel{-\!\!<} B & \text{iff} & \exists u \leq w \,.\, u \Vdash A \text{ and } u \dashV B \\
w \dashV A \mathrel{-\!\!<} B & \text{iff} & \forall u \leq w \,.\, u \dashV A \text{ or } u \Vdash B
\end{array}
$$

Figure 2.7: Forcing and rejecting of `BiInt`-formulae

Similarly, the law of non-contradiction holds for the `Int`-fragment of `BiInt`: the following formula is a theorem of `BiInt`, if A contains only the `Int`-connectives:

$$(A \wedge \neg A) \rightarrow B \tag{2.2.7}$$

2.2.4.4 Kripke semantics

We now review Rauszer's Kripke semantics for bi-intuitionistic logic [100]: it is the obvious combination of the `Int` and `DualInt` semantics we have already presented. A `BiInt` frame is a pair $\langle W, \leq \rangle$, where W is a non-empty set of worlds and $\leq$ is a reflexive and transitive binary accessibility relation. A `BiInt` model $M = \langle W, \leq, V \rangle$ is a `BiInt` frame $\langle W, \leq \rangle$ together with a valuation $V : Atoms \rightarrow 2^W$ which obeys *persistence*:

$$\forall u, w \in W. \forall p \in Atoms. \left(\text{if } w \leq u \text{ and } w \in V(p) \text{ then } u \in V(p)\right).$$

Definition 2.2.1. *Given a model $M = \langle W, \leq, V \rangle$, a world $w \in W$ and an atom $p \in Atoms$, we write $w \Vdash p$ (w forces p) iff $w \in V(p)$, and we write $w \dashV p$ (w rejects p) iff $w \notin V(p)$. We define forcing and rejecting of compound formulae by mutual recursion in Figure 2.7.*

From the semantics, it is clear that the connectives $\neg$ and $\sim$ can be derived from $\rightarrow$ and $-\!\!<$ respectively. Therefore from now on we restrict our attention to the connectives $\rightarrow$, $-\!\!<$, $\wedge$, $\vee$ only.

By induction on the length of a formula A, it follows that the persistence property also holds for formulae, and the reverse persistence property holds for formulae, that is:

Persistence: $\forall M = \langle W, \leq, V\rangle.\forall w \in W.\forall A \in Fml.\forall u \geq w.(\text{if } w \Vdash A \text{ then } u \Vdash A)$.

Reverse persistence: $\forall M = \langle W, \leq, V\rangle.\forall w \in W.\forall A \in Fml.\forall u \leq w.(\text{if } w \dashV A \text{ then } u \dashV A)$.

If we view the $\leq$ relation as describing the flow of time, then the persistence property says that once an atomic fact becomes known (true), it will remain known at all future points in time. Moreover, more atomic facts may become known as time progresses. Conversely, the reverse persistence property says that if an atomic fact is unknown (false) at the current point in time, it has been unknown at all past points in time. Moreover, more facts may become unknown as we revisit earlier points in time.

Definition 2.2.2. *Given a model $M = \langle W, \leq, V\rangle$ and a world $w \in W$, we write:*

$$w \Vdash \Gamma \quad \textit{iff} \quad \forall A \in \Gamma.w \Vdash A \qquad\qquad w \dashV \Delta \quad \textit{iff} \quad \forall A \in \Delta.w \dashV A.$$

Definition 2.2.3 (Validity and falsifiability)**.** *Given sets Γ and Δ of* `BiInt` *formulae, a* `BiInt` *formula A and a* `BiInt` *model $M = \langle W, \leq, V\rangle$, we write:*

$$\begin{array}{lll@{\qquad}lll} M \Vdash A & \textit{iff} & \forall w \in W.w \Vdash A & M \dashV A & \textit{iff} & \exists w \in W.w \dashV A \\ M \Vdash \Gamma & \textit{iff} & \forall w \in W.w \Vdash \Gamma & M \dashV \Delta & \textit{iff} & \exists w \in W.w \dashV \Delta. \end{array}$$

Then validity, falsifiability, satisfiability and unsatisfiability are:

$$\textit{validity} \qquad \forall M.M \Vdash A \tag{2.2.8}$$
$$\textit{falsifiability} \qquad \exists M.M \dashV A \tag{2.2.9}$$
$$\textit{satisfiability} \qquad \exists M = \langle W, \leq, V\rangle.\exists w \in W.w \Vdash A \tag{2.2.10}$$
$$\textit{unsatisfiability} \qquad \forall M = \langle W, \leq, V\rangle.\forall w \in W.w \dashV A. \tag{2.2.11}$$

We write $\vDash_{\texttt{BiInt}} A$ for "A is `BiInt`*-valid", and $\dashv_{\texttt{BiInt}} A$ for "A is* `BiInt`*-falsifiable".*

2.2.4.5 Rauszer's Hilbert calculus

Rauszer's Hilbert calculus for `BiInt` [99; 100] consists of the axioms given in Figure 2.8 and the following two inference rules:

$$\frac{A \to B \quad A}{B}\ MP \qquad\qquad \frac{A}{\neg \sim A}\ R$$

Axioms 2.2.21 to 2.2.29 relate the `Int` and `DualInt` connectives - we call these *interaction axioms*. We now give some intuitions (no pun intended) on the exclusion connective and interaction axioms. While implication $A \to B$ can be read as "if we can verify A, then we can verify B", exclusion $A \prec B$ can be read as "we can verify A but we do not have enough information to verify B". Then the intuition of axiom 2.2.21 in

$$(A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C)) \tag{2.2.12}$$
$$A \rightarrow A \vee B \tag{2.2.13}$$
$$B \rightarrow A \vee B \tag{2.2.14}$$
$$(A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C)) \tag{2.2.15}$$
$$(A \wedge B) \rightarrow A \tag{2.2.16}$$
$$(A \wedge B) \rightarrow B \tag{2.2.17}$$
$$(C \rightarrow A) \rightarrow ((C \rightarrow B) \rightarrow (C \rightarrow (A \wedge B))) \tag{2.2.18}$$
$$(A \rightarrow (B \rightarrow C)) \rightarrow ((A \wedge B) \rightarrow C) \tag{2.2.19}$$
$$((A \wedge B) \rightarrow C) \rightarrow (A \rightarrow (B \rightarrow C)) \tag{2.2.20}$$
$$A \rightarrow (B \vee (A \prec B)) \tag{2.2.21}$$
$$(A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A) \tag{2.2.22}$$
$$(A \prec B) \rightarrow \sim (A \rightarrow B) \tag{2.2.23}$$
$$((A \prec B) \prec C) \rightarrow (A \prec (B \vee C)) \tag{2.2.24}$$
$$\neg(A \prec B) \rightarrow (A \rightarrow B) \tag{2.2.25}$$
$$(A \rightarrow (B \prec B)) \rightarrow \neg A \tag{2.2.26}$$
$$\neg A \rightarrow (A \rightarrow (B \prec B)) \tag{2.2.27}$$
$$((B \rightarrow B) \prec A) \rightarrow \sim A \tag{2.2.28}$$
$$\sim A \rightarrow ((B \rightarrow B) \prec A) \tag{2.2.29}$$

Figure 2.8: Hilbert axioms for `BiInt`

Figure 2.8 is "if we can verify A, then we can either verify B, or we can verify the fact that (we can verify A but we do not have enough information to verify B)".

2.2.4.6 Sequent calculi

While the proof theory of intuitionistic logic and dual intuitionistic logic separately has been studied extensively and there are many cut-free sequent calculi for intuitionistic logic (e.g. [46; 36; 34]) and dual intuitionistic logic (e.g. [114]), the case for bi-intuitionistic logic is less satisfactory. Although Rauszer presented a sequent calculus for bi-intuitionistic logic and "proved" it cut-free [99], Pinto and Uustalu [95] have recently shown that it fails cut-elimination. Rauszer's sequent calculus is incomplete without cut because it does not handle the interaction between intuitionistic implication $\rightarrow$ and dual intuitionistic exclusion $-\!\!<$.

We now give an example that explains this interaction in more detail. Consider the sequent[4] $p \Rightarrow q, r \rightarrow ((p -\!\!< q) \wedge r)$ [95]. It has a derivation using cut in Rauszer's G1 [99], as shown below:

Example 2.2.4.

$$\dfrac{\dfrac{\dfrac{}{q \Rightarrow q}\,id \qquad \dfrac{}{p \Rightarrow p}\,id}{p \Rightarrow q, p -\!\!< q}\,-\!\!<_R \qquad \dfrac{\dfrac{\dfrac{}{p -\!\!< q, r \Rightarrow p -\!\!< q}\,id \qquad \dfrac{}{p -\!\!< q, r \Rightarrow r}\,id}{p -\!\!< q, r \Rightarrow (p -\!\!< q) \wedge r}\,\wedge_R}{p -\!\!< q \Rightarrow r \rightarrow ((p -\!\!< q) \wedge r)}\,\rightarrow_R}{p \Rightarrow q, r \rightarrow ((p -\!\!< q) \wedge r)}\,cut$$

The end sequent contains three complementary pairs, a positive and negative occurrence of p, q and r respectively: note that p occurs positively and q negatively in the $p -\!\!< q$ in the end sequent. In particular, the pairs involving p and q occur in the axioms and are essential for the derivation.

Now consider a backward attempt to derive this sequent without cut. The only non-structural rule that could give the conclusion is the $\rightarrow_R$ rule. But for intuitionistic soundness, the conclusion of $\rightarrow_R$ in Rauszer's calculus needs to be restricted to a singleton succedent. Thus we must weaken away the q to obtain the premise $p \Rightarrow r \rightarrow ((p -\!\!< q) \wedge r)$. But we have lost the essential occurrence of q. The only alternative is to weaken away p or $r \rightarrow ((p -\!\!< q) \wedge r)$. But neither of the resulting premises is derivable.

Crolard's dependency tracking [28] is one way to relax the intuitionistic restriction of "singletons on the right" to retain the essential q, so that cases like the above example remain cut-free derivable whilst retaining soundness. The idea is to record the dependencies between antecedents and succedents of the axioms, and use these dependencies to impose side conditions on the rules. Thus dependency tracking is not immediately suitable for backward proof search since the side conditions need to be known when the rules are applied backwards, before the axioms, and hence the dependencies, are computed.

[4]Using our previous notation, we would write $p_0 \Rightarrow q_0, r_0 \rightarrow ((p_0 -\!\!< q_0) \wedge r_0)$. However, from now on, we will leave out the subscripts for readability.

2.2.5 Tense logic

Having introduced BiInt, we now turn our attention to tense logic, which bears many similarities to BiInt, both from semantical and proof theoretic perspectives. Tense logic is an extension of modal logic with black box $\blacksquare$ and black diamond $\blacklozenge$ operators.

2.2.5.1 Syntax

Formally, formulas of tense logic are given by the following grammar, where p is an atom:

$$A := p \mid \top \mid \bot \mid \neg A \mid A \to A \mid A \wedge A \mid A \vee A \mid \Box A \mid \Diamond A \mid \blacksquare A \mid \blacklozenge A.$$

The axioms of minimal tense logic Kt are all the axioms of classical propositional logic, plus the following [85]:

$$\Box(A \to B) \to (\Box A \to \Box B) \tag{2.2.30}$$
$$\blacksquare(A \to B) \to (\blacksquare A \to \blacksquare B) \tag{2.2.31}$$
$$A \to \Box\blacklozenge A \tag{2.2.32}$$
$$A \to \blacksquare\Diamond A \tag{2.2.33}$$

The theorems of Kt are those that are generated from the above axioms and their substitution instances using the following rules:

$$\frac{A \to B \quad A}{B}\, MP \qquad \frac{A}{\Box A}\, Nec\Box \qquad \frac{A}{\blacksquare A}\, Nec\blacksquare$$

As can be seen from axioms 2.2.32 and 2.2.33, there is an interaction between the $\Box$ and $\blacklozenge$ connectives of tense logic, as well as between the $\blacksquare$ and $\Diamond$ connectives. This interaction makes the development of a cut-free sequent calculus non-trivial, just as in the case of bi-intuitionistic logic, where there is an interaction between $\to$ and $-\!\!<$.

2.2.5.2 Kripke semantics

Semantically, Kt can be defined using Kripke frames with bi-directional relations. More formally: A Kt-*frame* is a pair $\langle W, R\rangle$, with W a non-empty set (of worlds) and $R \subseteq W \times W$. A Kt-model is a triple $\langle W, R, V\rangle$, with $\langle W, R\rangle$ a Kt frame and $V : Atoms \to 2^W$ a valuation mapping each atom to the set of worlds where it is true.

For a world $w \in W$ and an atom $p \in Atoms$, if $w \in V(p)$ then we write $w \Vdash p$ and say p is forced by w; otherwise we write $w \nVdash p$ and say p is rejected by w. Forcing of compound formulae is defined by mutual recursion in Figure 2.9. A Kt-formula A is valid if and only if it is forced by all worlds in all models, i.e. if and only if $w \Vdash A$ for all $\langle W, R, V\rangle$ and for all $w \in W$. A Kt-formula A is falsifiable if and only if it is rejected

$w \Vdash \top$		for all w	$w \nVdash \bot$		for all w
$w \Vdash \neg A$	iff	$w \nVdash A$	$w \Vdash A \rightarrow B$	iff	$w \nVdash A$ or $w \Vdash B$
$w \Vdash A \vee B$	iff	$w \Vdash A$ or $w \Vdash B$	$w \Vdash A \wedge B$	iff	$w \Vdash A$ and $w \Vdash B$
$w \Vdash \Box A$	iff	$\forall u.$ if wRu then $u \Vdash A$	$w \Vdash \Diamond A$	iff	$\exists u.wRu$ and $u \Vdash A$
$w \Vdash \blacksquare A$	iff	$\forall u.$ if uRw then $u \Vdash A$	$w \Vdash \blacklozenge A$	iff	$\exists u.uRw$ and $u \Vdash A$

Figure 2.9: Forcing of `Kt`-formulae

by some world in some model, i.e. if and only if $w \nVdash A$ for some $\langle W, R, V\rangle$ and for some $w \in W$.

Viewed philosophically, tense logic allows to reason about the flow of time [85]: while the $\Box$ and $\Diamond$ modalities refer to future worlds, the $\blacksquare$ and $\blacklozenge$ modalities refer to past worlds.

Just as the basic modal logic `K` can be extended by various axioms that semantically correspond to frame conditions, we can build extensions of the minimal tense logic `Kt`. For example, we can obtain the reflexive-transitive extension `Kt.S4` by adding the following axioms:

$$4_{\Box} : \quad \Box A \rightarrow \Box\Box A \tag{2.2.34}$$

$$4_{\blacksquare} : \quad \blacksquare A \rightarrow \blacksquare\blacksquare A \tag{2.2.35}$$

$$T_{\Diamond} : \quad A \rightarrow \Diamond A \tag{2.2.36}$$

$$T_{\blacklozenge} : \quad A \rightarrow \blacklozenge A \tag{2.2.37}$$

2.2.6 Relationship between bi-intuitionistic logic and tense logic

In this section, we highlight the similarities between `BiInt` and tense logic with reflexive transitive frames `Kt.S4` by reviewing two translations between these logics.

2.2.6.1 Wolter's translation from `BiInt` to `Kt.S4`

Starting with Gödel [48], many authors have developed translations from `Int`-formulae to `S4`-formulae that preserve validity, that is, they map valid `Int`-formulae to valid `S4`-formulae, and falsifiable `Int`-formulae to falsifiable `S4`-formulae. Using the usual symmetry between `Int` and `DualInt`, `Int`-to-`S4` translations can be extended to give translations from `BiInt` to `Kt.S4`. The following translation has been developed by Wolter [121]; an equivalent translation was presented by Łukowski [82], but his paper contains a typographical error.

Definition 2.2.5 (`BiInt` to `Kt.S4` translation)**.** *In the following, q is an atomic formula, A* *and B are* `BiInt` *formulae, and $T(.)$ is a* `Kt.S4` *formula.*

$$\begin{array}{lcl} T(q) & = & \Box q \\ T(A \wedge B) & = & T(A) \wedge T(B) \\ T(A \vee B) & = & T(A) \vee T(B) \\ T(\neg A) & = & \Box \neg T(A) \\ T(A \rightarrow B) & = & \Box(T(A) \rightarrow T(B)) \\ T(\sim A) & = & \blacklozenge \neg T(A) \\ T(A \prec B) & = & \blacklozenge(T(A) \wedge \neg T(B)) \end{array}$$

Theorem 2.2.6. *The decision problem for* `BiInt` *is* PSPACE*-complete.*

Proof. To show that `BiInt` is in PSPACE, we use the polynomial translation of `BiInt` into `Kt.S4` given in Definition 2.2.5. Since `Kt.S4` is in PSPACE [105], we know that `BiInt` is also in PSPACE.

To show that `BiInt` is PSPACE-hard, we use the fact that `BiInt` is an extension of `Int`, which is PSPACE-complete [106], and hence PSPACE-hard. Therefore `BiInt` is PSPACE-complete. Q.E.D.

Although we won't be using it in the rest of this thesis, there is a reverse translation from `S4` to `Int` which we briefly review next.

2.2.6.2 Fernandez's translation from `S4` to `Int`

Fernandez [41] has developed a polynomial validity-preserving translation from `S4`-formulae to `Int`-formulae; see also earlier unpublished work by Egly [37]. As both authors point out, the translation from `S4`-formulae to `Int`-formulae is much more difficult than the reverse direction. Fernandez explains this difficulty semantically: while `S4` models consist of Kripke frames which contain clusters of worlds (a cluster is a set if worlds C such that for any $u, w \in C$, it is the case that uRw), `Int` frames are trees. Additionally, `S4` obeys the law of the excluded middle, while `Int` does not. The basic idea of Fernandez's translation is to use a "layering" scheme to simulate the clusters of `S4` models in `Int`. He adds a number of new propositional atoms to simulate $\Box$-formulae.

We conjecture that it is possible to extend Fernandez's translation to a translation from `Kt.S4`-formulae to `BiInt`-formulae: using the symmetry between `Int` and `DualInt`, we would add special propositional atoms to simulate $\blacksquare$-formulae of `Kt.S4` similarly to Fernandez's translation of $\Box$-formulae.

2.3 Motivation

As discussed in Section 2.2.4.6, Rauszer's sequent calculus for bi-intuitionistic logic is incomplete without the cut rule. As illustrated by Uustalu's counterexample, traditional sequent calculi methods fail for bi-intuitionistic logic. This problem is not unique to bi-intuitionistic logic; it is also present in a range of other non-classical logics (e.g. [5; 7; 25]). As a result, a range of other extended sequent mechanisms have been developed that give cut-free sequent calculi for complicated logics where traditional sequent calculi fail.

These extended sequent mechanisms differ from traditional sequent mechanisms in terms of the structures contained in the sequents, as well as the kind of rules that can be applied to the sequents. For example, a *hypersequent* [5; 7; 25] consists of one or more traditional sequents, while a *nested sequent* is a tree of traditional sequents [87; 88; 35; 73; 17; 19; 97].

The aim of this thesis is to develop extended sequent calculi mechanisms for bi-intuitionistic and tense logic, as well as the combined bi-intuitionistic tense logic. We will do so using two complementary approaches which we introduce next.

2.3.1 A calculus of derivations and refutations

In Part I (Chapter 3) of this thesis, we will give a purely syntactic cut-free sequent calculus for bi-intuitionistic logic which combines derivations and refutations as first-class citizens. We now set the scene by reviewing the notions of traditional derivation calculi, theorems, refutation calculi, non-theorems, proof/refutation search and counter-models, and explaining why a combined calculus makes sense.

Derivation calculi are used to reason about a syntactic derivability relation $\vdash$. For example, Gentzen's LJ [46] is a sequent calculus for propositional intuitionistic logic, where a judgement $\vdash \Gamma \Rightarrow A$ means the sequent $\Gamma \Rightarrow A$ is derivable: that is, the formula A is syntactically derivable from the multiset of formulae Γ in intuitionistic logic. Increasingly, sequent calculi are used to decide whether $\Gamma \Rightarrow A$ is derivable by applying the rules backwards, so it has become important to study such calculi from this "*proof-search*" perspective. Indeed, a "contraction-free" variant of LJ, called LJT [36] can be used for proof-search in intuitionistic logic. But note that a single non-derivation is not really a first-class citizen in this setting.

Refutation calculi are syntactic formalisms for reasoning about a syntactic refutability relation $\dashv$ (say). They show syntactically that a formula is a non-theorem and were introduced to modern logic by Łukasiewicz [81], although the idea originated from Aristotle. For example, Goranko has given refutation calculi for some modal logics [50] where the judgement $\dashv A$ means that A is refutable, i.e., a non-theorem of the logic. The notion of "backward refutation-search" asks whether A is refutable under the assumptions in Γ, and some refutation calculi have been designed with this aim. For example, using $\hat{\Gamma}/\check{\Delta}$ as a conjunction/disjunction of all the members of Γ/Δ, Pinto and Dyckhoff use "sequents" of the form $\Gamma \not\Rightarrow \Delta$ to give a refutation "sequent" calculus CRIP for intuitionistic logic [94] where the judgement $\vdash_{CRIP} \Gamma \not\Rightarrow \Delta$ means that the formula $\hat{\Gamma} \to \check{\Delta}$ is a non-theorem. Importantly, these calculi produce refutations that are first-class objects (trees).

As usual, we can relate syntactic derivability (in LJT) to semantics if the calculus is sound and complete: thus $\vdash \Gamma \Rightarrow A$ (in LJT) iff $\hat{\Gamma} \to A$ is *valid* (in intuitionistic logic). Such a correspondence is vital in many applications: we pinpoint why A is not derivable from Γ by constructing a *counter-model* showing that $\hat{\Gamma} \to A$ is *falsifiable*. But since derivation calculi do not construct counter-models directly, the counter-model is constructed using meta-level reasoning to "stitch" together many non-derivations of the sequent $\Gamma \Rightarrow A$.

Dually, we can relate syntactic refutability (in CRIP) to semantics if the calculus is sound and complete: thus $\vdash \Gamma \not\Rightarrow A$ (in CRIP) iff the formula $\hat{\Gamma} \to A$ is *falsifiable* (in intuitionistic logic). Indeed, specially designed refutation calculi, such as CRIP, allow us to reason about refutability *and* obtain a counter-model since a single refutation corresponds directly to a counter-model. Of course, they are not immediately suitable for demonstrating validity.

Although derivation calculi and refutation calculi are usually studied as distinct calculi, there are desirable *meta-level* relationships between derivability and refutability (for the same logic). For example, for any input Γ and A, either there is a derivation of $\Gamma \Rightarrow A$ in LJT, or a refutation of $\Gamma \not\Rightarrow A$ in CRIP [94]. It therefore makes sense to ask what would happen if we were to combine derivation calculi with refutation calculi in one single setting. For example, the *modus tollens* rule used in some refutation calculi *combines* a *derivation* of $A \to B$ and a *refutation* of B to obtain a *refutation* of A. As Goranko suggests, we could also combine *derivations* and *refutations* to produce *derivations*. Indeed, he predicts that such *combined deductive systems* "have a greater potential efficiency than the orthodox ones, since they can employ on a syntactic level self-reference to some of their meta-features, which are beyond the expressive abilities of the traditional systems" [50].

To retain the link with semantics as well as the potential for backward (proof or refutation) search, the combined calculus must be such that derivations/refutations preserve validity / counter-models downwards while providing a decision procedure if our logic is decidable. There is a subtlety here, for Larchey-Wendling [80] has already combined proof search and explicit counter-model construction to obtain an efficient decision procedure for an extension of intuitionistic logic called Gödel-Dummett logic. But Larchey-Wendling constructs a counter-model merely as a tool used at certain times during proof search. Thus his calculus does not contain derivations and refutations as first-class citizens.

2.3.2 Nested sequent calculi

In Part II of this thesis, we consider the broader problem of proof search in display calculi (introduced shortly), and use display-like calculi to develop reasoning techniques for bi-intuitionistic logic (Chapters 4 and 5), tense logic (Chapter 6) and a combination of the two logics (Chapter 7).

Belnap's Display Logic [12] (we prefer the term display calculi) is an extremely general proof-theoretical framework; see also Chapter A for a more detailed introduction to display calculi. A display calculus obeys the *display property*: any sequent containing a particular formula occurrence A can be transformed into another sequent in which the occurrence of A is either the whole of the antecedent or the whole of the succedent, using only a subset of the rules called the *display postulates*. The occur-

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overset{\vdots}{A \Rightarrow B > C}}{A \Rightarrow C, (B > C)}\, w_R}{(A < (B > C)) \Rightarrow C}\, rp_<}{(A < (B > C)), B \Rightarrow C}\, w_L}{A < (B > C) \Rightarrow B > C}\, rp_>}{A \Rightarrow (B > C), (B > C)}\, rp_<}{A \Rightarrow B > C}\, c_R}{A \Rightarrow B \rightarrow C}\, \rightarrow_R$$

Figure 2.10: Non-terminating backward proof search attempt in a display calculus for bi-intuitionistic logic

rence of A is then said to be displayed. The most pleasing property of display calculi however is that if the rules of the display calculus enjoy eight easily checked conditions, then the calculus is guaranteed to obey cut-admissibility. That is, one single cut-admissibility proof suffices for all display calculi. This modularity makes it an excellent framework for designing sequent calculi for logics, particularly when we wish to mix and match the intuitionistic, modal, or substructural aspects of different logics into a new logic [116; 51]. In particular, Goré has developed a display calculus for bi-intuitionistic logic [53], and Wansing has recently extended a variant of Goré's system with constructive negation [120].

The generality of display calculi is obtained by adding a structural proxy for every logical connective and using *residuation principles* to implement the display property. For example, a display calculus for bi-intuitionistic logic contains Gentzen's "comma", but also two binary structural connectives "$>$" and "$<$" which allow us to hide structures by nesting them inside one another. The following are the logical rules for implication and exclusion in Goré's display calculus for bi-intuitionistic logic [53]:

$$\dfrac{X \Rightarrow A \qquad B \Rightarrow Y}{A \rightarrow B \Rightarrow X > Y}\, \rightarrow_L \qquad \dfrac{Z \Rightarrow A > B}{Z \Rightarrow A \rightarrow B}\, \rightarrow_R$$

$$\dfrac{A < B \Rightarrow Z}{A \prec B \Rightarrow Z}\, \prec_L \qquad \dfrac{A \Rightarrow X \qquad Y \Rightarrow B}{X < Y \Rightarrow A \prec B}\, \prec_R$$

Here $>$ is a structural proxy for $\rightarrow$, and $<$ is a structural proxy for $\prec$.

The following are the structural rules for the connectives "$>$" and "$<$", where $rp_>$ implements residuation between comma and $>$, and $rp_<$ implements residuation between comma and $<$, and double lines indicate that the rule may be used both reading from top to bottom and vice versa:

$$\begin{array}{c} X, Y \Rightarrow Z \\ \hline\hline Y \Rightarrow X > Z \end{array}\, rp_> \qquad \begin{array}{c} Z \Rightarrow X, Y \\ \hline\hline Z < Y \Rightarrow X \end{array}\, rp_<$$

The main disadvantage of display calculi is that the display postulates can and must create large structures during the process of displaying a particular formula occurrence, making display calculi bad for backward proof-search. More specifically, the invertible structural display postulate rules (for example, $rp_>$ and $rp_<$ above) allow "pointless" shuffling of structures and easily lead to non-termination of proof search if applied naively. These rules are at the heart of display calculi and guarantee the display property, therefore eliminating them without losing the display property is not obvious.

Another issue is the presence of explicit contraction and weakening rules in display calculi which are couched in terms of structures rather than formulae. Replacing these rules with ones based on formulae can break one of the conditions for a display calculus, namely, the (C6/C7) condition that "each rule is closed under simultaneous substitution of arbitrary structures for congruent parameters" [75]. Absorbing them completely to obtain a "contraction-free" calculus is thus not an obvious step. Figure 2.10 illustrates both problems.

To sum up, a disciplined proof-theoretic methodology for transforming a display calculus into a more manageable traditional "contraction-free" and "residuation rule free" calculus whilst preserving cut-admissibility is an important goal. Although display calculi were not designed for automated proof-search there is a surprising lack of interest in the study of proof search for display logics: the only exceptions are the works of Wansing [117] and Restall [101].

Our first step towards taming display calculi is to limit the structural connectives used in the calculi and consequently, the number of display postulates. Specifically, we work within display structures which can be viewed as a tree of traditional Gentzen's sequents, called *nested sequents*, which have been used previously by Kashima [73] and, independently, by Brünnler [17; 18] and Poggiolesi [97] to present several modal and tense logics. We comment further on the various nested sequent calculi in Section 8.2.

Nested sequent calculi allow either "shallow" or "deep" inference: in shallow inference calculi inference rules are applied at the top/root level only, and residuation rules are used to re-orient the trees to bring the required structures to the top-level. In deep inference calculi, inference rules can be applied at any level, and propagation rules move formulae around the trees. "Deep" inference is a refinement of shallow inference, since we do need to bring the required structure to the top-level to apply a rule to a formula in this structure, but we can simply apply the rule to the formula inside the structure.

Since residuation rules are largely responsible for the difficulty in finding a proof search procedure for display-like calculi, our second step is therefore to eliminate these residuation rules without losing completeness; and we will do so by using deep inference. More precisely, we will show that we can simulate residuation using deep inference for a range of logics: bi-intuitionistic logic (chapter 5) and tense logic (chapter 6), and finally bi-intuitionistic tense logic (chapter 7), as well as their sub-logics and some extensions. Note that the idea of using deep inference for taming proof search is not entirely new: Areces and Bernardi [3] appear to be the first to have noticed the connection between deep inference and residuation in display logic in the context of categorial grammar. However, they do not give an explicit proof of this correspondence as we do here for our calculi.

2.4 Conclusion

In this chapter, we have reviewed the concepts of syntax, semantics, proof calculi and proof search in the context classical propositional logic. Additionally, we introduced a number of object logics that we will use throughout the rest of the thesis: modal, intuitionistic, dual-intuitionistic, bi-intuitionistic and, finally, tense logic. We then discussed the limitations of traditional proof search methods for logics such as bi-intuitionistic and tense logic, and introduced the two approaches we will use to address these limitations.

Part I

Bi-intuitionistic logic

Chapter 3

A calculus of derivations and refutations for bi-intuitionistic logic

In this chapter, we give a purely syntactic cut-free sequent calculus for bi-intuitionistic logic which combines derivations and refutations as first-class citizens. In particular, both soundness and completeness are proved in a purely top-down manner. This is the first cut-free and complete sequent calculus for bi-intuitionistic logic. Moreover, it can be used for backward proof search. We remind the reader that the syntax and semantics of bi-intuitionistic logic were described in Section 2.2.4.

In section 3.1, we give a high-level overview of our sequent calculus and show an example derivation that illustrates our calculus, before we formally present the calculus **GBiInt** in section 3.2. We prove the soundness and completeness of **GBiInt** in section 3.3. In section 3.4, we describe a decision procedure for bi-intuitionistic logic and analyse its computational complexity. In section 3.5, we compare **GBiInt** to previous work.

Note. The results of this chapter have been published in [22] and [55].

3.1 An overview of GBiInt

A **GBiInt** sequent is an expression $\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P}$ or $\mathcal{S}\ \Gamma\ \not\Rightarrow\ \Delta\ \mathcal{P}$ where Γ/Δ are traditional sets of formulae, $\mathcal{S}/\mathcal{P}$ are sets of sets of formulae, and the turnstiles $\Rightarrow$ and $\not\Rightarrow$ indicate whether we have a derivation or a refutation. The extra components $\mathcal{S}/\mathcal{P}$ are variables, which are a mechanism to pass information from premises to conclusions [103], similarly to attributes in attribute grammars [74]. In our case, the variables are sets of sets of formulae containing subformulae discovered at the leaves of refutation trees. Our rules transmit these essential formulae down towards the root of refutations, and use these formulae to obtain a derivation from a refutation. In fact, we obtain a demand-driven cut as viewed from a backward (proof/refutation) search perspective: rather than having to guess cut formulae at each sequent, we perform cut-free backward search as usual, and use the contents of variables when we find a refutation: Section 3.2 gives details.

We then relate our generalised syntactic judgement $\overset{?}{\Rightarrow}$ (either $\Rightarrow$ or $\not\Rightarrow$) to a generalised semantic judgement (either validity or falsifiability) via a combined soundness and completeness proof. We show that the rules preserve the generalised semantic judgement downwards: validity/falsifiability is preserved downwards in derivation/refutation trees. Thus derivable sequents are valid and refutable sequents are falsifiable. Additionally, we show that in certain special cases we can combine a refutation with a derivation to obtain a derivation. Finally, we give a terminating procedure which decides whether a generalised sequent $\Gamma \overset{?}{\Rightarrow} \Delta$ is derivable or refutable using a number of side conditions that the rules must obey. Thus completeness follows directly from our ability to derive or refute every input sequent, rather than indirectly from the failure of a systematic proof search procedure.

We now give a high-level overview of how our sequent calculus **GBiInt** solves the difficulties posed by interaction formulae in bi-intuitionistic logic that we discussed in section 2.2.4.

Recall that the sequent $p \Rightarrow q, r \rightarrow ((p \prec q) \wedge r)$ has a derivation using cut in Rauszer's calculus, but does not have a cut-free derivation [95]. A cut-free derivation of this sequent in our calculus **GBiInt** ends as shown below:

$$\dfrac{\overset{\cdots}{\{\cdots\}\; p, r \not\Rightarrow (\mathbf{p} \prec \mathbf{q}) \wedge r\; \{\{\mathbf{p} \prec \mathbf{q}\}\}} \qquad \overset{\cdots}{p \Rightarrow \mathbf{q}, r \rightarrow ((\mathbf{p} \prec \mathbf{q}) \wedge r), \mathbf{p} \prec \mathbf{q}}}{p \Rightarrow \mathbf{q}, r \rightarrow ((p \prec q) \wedge r)} \rightarrow_{R2}$$

In addition to the traditional antecedent and succedent, our sequents contain two non-traditional components, which we call variables. In the derivation sketch above, $\{\{p \prec q\}\}$ is one of the variables of the top left sequent; we have not shown the variables of the other sequents. The "$\not\Rightarrow$" turnstile in the left premise denotes a refutation, the "$\Rightarrow$" turnstile in the right premise denotes a derivation, and the $\rightarrow_{R2}$ rule allows us to compose the refutation with a derivation to produce a derivation.

Notice that the variable $\{\{p \prec q\}\}$ in the left premise contains the crucial subformula $p \prec q$, which is also present in the formula part of the right premise. This is a crucial feature of our calculus; it can also be viewed in an operational way as a "flow" of variables from a refutation in the left premise of the $\rightarrow_{R2}$ rule to the derivation in the right premise of the rule.

3.2 The sequent calculus

We now present a Gentzen-style sequent calculus for `BiInt`. The sequents have a non-traditional component in the form of variables that are sets of sets of formulae. When our calculus is used for backward search, the variables are instantiated at certain leaves of the search tree, and passed to lower sequents from premises to conclusion. Note that the variables are not names for Kripke worlds, so our sequents contain no semantic features.

3.2.1 Sequents

We introduce an extended syntax to simplify the presentation of some of our sequent rules.

Definition 3.2.1. *If A is a* `BiInt` *formula, then A is an extended* `BiInt` *formula. If $\mathcal{Q}$ is a set $\{\{A_0^0, \cdots, A_0^{n_0}\}, \cdots, \{A_m^0, \cdots, A_m^{n_m}\}\}$ of sets of* `BiInt` *formulae, then $\bigvee \mathcal{Q}$ and $\bigwedge \mathcal{Q}$ are extended* `BiInt` *formulae with intended semantics*

$$\bigvee \mathcal{Q} \equiv (A_0^0 \wedge \cdots \wedge A_0^{n_0}) \vee \cdots \vee (A_m^0 \wedge \cdots \wedge A_m^{n_m}) \tag{3.2.1}$$

$$\bigwedge \mathcal{Q} \equiv (A_0^0 \vee \cdots \vee A_0^{n_0}) \wedge \cdots \wedge (A_m^0 \vee \cdots \vee A_m^{n_m}). \tag{3.2.2}$$

The following semantics follows directly from Definition 3.2.1:

Definition 3.2.2. *Given a* `BiInt` *model $M = \langle W, \leq, V\rangle$, a world $w \in W$, and two extended* `BiInt` *formulae $\bigvee \mathcal{S}$ and $\bigwedge \mathcal{P}$, we write:*

$$\begin{array}{llll} w \Vdash \bigvee \mathcal{S} & \textit{iff} & \exists \Sigma \in \mathcal{S}. \forall A \in \Sigma. w \Vdash A & w \dashv\!\mid \bigvee \mathcal{S} & \textit{iff} & \forall \Sigma \in \mathcal{S}. \exists A \in \Sigma. w \dashv\!\mid A \\ w \Vdash \bigwedge \mathcal{P} & \textit{iff} & \forall \Pi \in \mathcal{P}. \exists A \in \Pi. w \Vdash A & w \dashv\!\mid \bigwedge \mathcal{P} & \textit{iff} & \exists \Pi \in \mathcal{P}. \forall A \in \Pi. w \dashv\!\mid A. \end{array}$$

We can now extend Definition 2.2.2 of forcing of sets of `BiInt` *formulae to forcing of sets of extended* `BiInt` *formulae in the obvious way. That is, if Γ and Δ are sets of extended* `BiInt` *formulae, and A is an extended* `BiInt` *formula, then:*

$$w \Vdash \Gamma \quad \textit{iff} \quad \forall A \in \Gamma. w \Vdash A \qquad\qquad w \dashv\!\mid \Delta \quad \textit{iff} \quad \forall A \in \Delta. w \dashv\!\mid A$$

Definition 3.2.3 (Sequent)**.** *A* **GBiInt** *sequent/antisequent is an expression of one of the forms*

$$\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P} \qquad\qquad \mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P}$$

and consists of the following components: a left hand side (LHS) Γ *which is a set of extended* `BiInt` *formulae; a* right hand side (RHS) Δ *which is a set of extended* `BiInt` *formulae; two* variables $\mathcal{S}$, $\mathcal{P}$, *each of which is a set of sets of* `BiInt` *formulae; and a* turnstile *which is either $\Rightarrow$ for traditional sequents or $\not\Rightarrow$ for antisequents.*

We shall often use the following simplifications when referring to sequents:

$\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow}\ \Delta\ \mathcal{P}$ when we are referring to either a traditional sequent or an antisequent;

$\Gamma \Rightarrow \Delta$ or $\Gamma \not\Rightarrow \Delta$ or $\Gamma \overset{?}{\Rightarrow} \Delta$ when the values of the variables are not important.

We now define the semantics of a sequent. In the following, τ is a translation from sequents to `BiInt` formulae, and σ is a translation from sequents to semantic judgements, and $\hat{\Gamma}/\check{\Delta}$ is a conjunction/disjunction of all the members of Γ/Δ:

$$\tau(\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow}\ \Delta\ \mathcal{P}) \ = \ \bigvee \mathcal{S} \wedge \hat{\Gamma} \to \check{\Delta} \vee \bigwedge \mathcal{P} \tag{3.2.3}$$

$$\sigma(\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P}) \ = \ \models_{\texttt{BiInt}} \tau(\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P}) \tag{3.2.4}$$

$$\sigma(\mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P}) \ = \ =\!\mid_{\texttt{BiInt}} \tau(\mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P}) \tag{3.2.5}$$

The reading of (3.2.4)/(3.2.5) is that the formula corresponding to a sequent/antisequent is valid/falsifiable.

Definition 3.2.4. *A sequent* $\Gamma \overset{?}{\Rightarrow} \Delta$ *is* saturated *iff all of the following hold:*

$$\begin{array}{ll} \multicolumn{2}{c}{\Gamma \cap \Delta = \emptyset} \\ \multicolumn{2}{c}{\Gamma \textit{ and } \Delta \textit{ contain only } \texttt{BiInt} \textit{ formulae}} \\ \textit{if } A \wedge B \in \Gamma \textit{ then } A \in \Gamma \textit{ and } B \in \Gamma & \textit{if } A \wedge B \in \Delta \textit{ then } A \in \Delta \textit{ or } B \in \Delta \\ \textit{if } A \vee B \in \Gamma \textit{ then } A \in \Gamma \textit{ or } B \in \Gamma & \textit{if } A \vee B \in \Delta \textit{ then } A \in \Delta \textit{ and } B \in \Delta \\ \textit{if } A \rightarrow B \in \Gamma \textit{ then } A \in \Delta \textit{ or } B \in \Gamma & \textit{if } A \prec B \in \Delta \textit{ then } A \in \Delta \textit{ or } B \in \Gamma \\ \textit{if } A \prec B \in \Gamma \textit{ then } A \in \Gamma & \textit{if } A \rightarrow B \in \Delta \textit{ then } B \in \Delta. \end{array}$$

Definition 3.2.5. *A sequent* $\Gamma \overset{?}{\Rightarrow} \Delta$ *is* strongly saturated *iff all of the following hold:*

(i) $\Gamma \overset{?}{\Rightarrow} \Delta$ *is saturated* (ii) *if* $A \rightarrow B \in \Delta$ *then* $A \in \Gamma$ (iii) *if* $A \prec B \in \Gamma$ *then* $B \in \Delta$.

3.2.2 Sequent rules and various calculi

We now describe the sequent rules, axioms and anti-axioms that are used to build derivation and refutation trees. Rather than summarising all the rules in one large table, we break them into groups and describe each group in turn. We start with the *axioms* and *anti-axiom*, which are the leaves of derivations and refutations respectively:

Axioms: $\emptyset\ \Gamma, A \ \Rightarrow\ \Delta, A\ \emptyset\ \ id$ $\emptyset\ \bot, \Gamma \ \Rightarrow\ \Delta\ \emptyset\ \ \bot_L$ $\emptyset\ \Gamma \ \Rightarrow\ \Delta, \top\ \emptyset\ \ \top_R$

Anti-axiom: $\{\Gamma\}\ \Gamma \ \not\Rightarrow\ \Delta\ \{\Delta\}$ Ret where $\Gamma \not\Rightarrow \Delta$ is strongly saturated.

The *axioms* *id*, $\bot_L$ and $\top_R$ have the traditional sequent turnstile "$\Rightarrow$", while the *anti-axiom* Ret has the antisequent turnstile "$\not\Rightarrow$". The rules will propagate these turnstiles down the trees, eventually arriving at the root, which will be a derivation if the root sequent is a traditional sequent ("$\Rightarrow$"), or a refutation if the root sequent is an antisequent ("$\not\Rightarrow$"). Note that the anti-axiom Ret instantiates the values of the $\mathcal{S}$ and $\mathcal{P}$ variables to $\{\Gamma\}$ and $\{\Delta\}$ respectively, while the axioms *id*, $\bot_L$ and $\top_R$ set the variables to empty sets $\emptyset$. The sequent rules will transmit the variables down the trees, and combine variables from multiple premises in some cases.

Using terminology from [52], the *static rules* of our sequent calculus are:

***α*-rules:**

$$\frac{\mathcal{S}\ \Gamma, A \wedge B, A, B \ \overset{?}{\Rightarrow}_1\ \Delta\ \mathcal{P}}{\mathcal{S}\ \Gamma, A \wedge B \ \overset{?}{\Rightarrow}_0\ \Delta\ \mathcal{P}}\ \wedge_L \qquad \frac{\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow}_1\ \Delta, A \vee B, A, B\ \mathcal{P}}{\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow}_0\ \Delta, A \vee B\ \mathcal{P}}\ \vee_R$$

$$\frac{\mathcal{S}\ \Gamma, A \prec B, A \ \overset{?}{\Rightarrow}_1\ \Delta\ \mathcal{P}}{\mathcal{S}\ \Gamma, A \prec B \ \overset{?}{\Rightarrow}_0\ \Delta\ \mathcal{P}}\ \prec_L^I \qquad \frac{\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow}_1\ \Delta, A \rightarrow B, B\ \mathcal{P}}{\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow}_0\ \Delta, A \rightarrow B\ \mathcal{P}}\ \rightarrow_R^I$$

Where $\overset{?}{\Rightarrow}_1 = \overset{?}{\Rightarrow}_0 \in \{\Rightarrow, \not\Rightarrow\}$.

***β*-rules:**

$$\frac{\mathcal{S}_1\ \Gamma, A \vee B, A\ \overset{?}{\Rightarrow}_1\ \Delta\ \mathcal{P}_1 \qquad \mathcal{S}_2\ \Gamma, A \vee B, B\ \overset{?}{\Rightarrow}_2\ \Delta\ \mathcal{P}_2}{\mathcal{S}_1 \cup \mathcal{S}_2\ \Gamma, A \vee B\ \overset{?}{\Rightarrow}_0\ \Delta\ \mathcal{P}_1 \cup \mathcal{P}_2} \vee_L$$

$$\frac{\mathcal{S}_1\ \Gamma\ \overset{?}{\Rightarrow}_1\ \Delta, A \wedge B, A\ \mathcal{P}_1 \qquad \mathcal{S}_2\ \Gamma\ \overset{?}{\Rightarrow}_2\ \Delta, A \wedge B, B\ \mathcal{P}_2}{\mathcal{S}_1 \cup \mathcal{S}_2\ \Gamma\ \overset{?}{\Rightarrow}_0\ \Delta, A \wedge B\ \mathcal{P}_1 \cup \mathcal{P}_2} \wedge_R$$

$$\frac{\mathcal{S}_1\ \Gamma, A \to B\ \overset{?}{\Rightarrow}_1\ \Delta, A\ \mathcal{P}_1 \qquad \mathcal{S}_2\ \Gamma, A \to B, B\ \overset{?}{\Rightarrow}_2\ \Delta\ \mathcal{P}_2}{\mathcal{S}_1 \cup \mathcal{S}_2\ \Gamma, A \to B\ \overset{?}{\Rightarrow}_0\ \Delta\ \mathcal{P}_1 \cup \mathcal{P}_2} \to_L$$

$$\frac{\mathcal{S}_1\ \Gamma, B\ \overset{?}{\Rightarrow}_1\ \Delta, A \prec B\ \mathcal{P}_1 \qquad \mathcal{S}_2\ \Gamma\ \overset{?}{\Rightarrow}_2\ \Delta, A \prec B, A\ \mathcal{P}_2}{\mathcal{S}_1 \cup \mathcal{S}_2\ \Gamma\ \overset{?}{\Rightarrow}_0\ \Delta, A \prec B\ \mathcal{P}_1 \cup \mathcal{P}_2} \prec_R$$

$$\text{Where } \overset{?}{\Rightarrow}_0 = \begin{cases} \Rightarrow & \text{if } \overset{?}{\Rightarrow}_1 = \Rightarrow \text{ and } \overset{?}{\Rightarrow}_2 = \Rightarrow \\ \not\Rightarrow & \text{otherwise.} \end{cases}$$

These rules use many features of Dragalin's `GHPC` [34] for intuitionistic logic (`Int`); we have added symmetric rules for the dual intuitionistic logic (`DualInt`) connective $\prec$. We chose Dragalin's *multi-succedent* calculus since the restriction to single succedents/antecedents for some sequents is one of the causes of incompleteness in Rauszer's calculus for `BiInt` [99]; see also Maehara [83] for early work on a multi-succedent calculus for `Int`. But using Dragalin's calculus and its dual does not give us `BiInt` completeness. We therefore also follow Schwendimann's approach [103] of passing relevant information from premises to conclusions using variables, which we instantiate at the refutation leaves: see Ret above.

We have also added the static rule $\to_R^I$ for implication on the right (and symmetrically, $\prec_L^I$) originally given by Śvejdar [107]. Although Śvejdar himself does not give the semantics behind this rule, or explain the precise role it plays in his calculus, his rules are best explained by reading them from conclusion to premises, as used in backward search. Consider the $\to_R^I$ rule when $\overset{?}{\Rightarrow}_0 = \overset{?}{\Rightarrow}_1 = \not\Rightarrow$. As we shall show later, finding a refutation of the conclusion involves falsifying the formula $A \to B$. Rather than immediately creating the successor that falsifies $A \to B$, the $\to_R^I$ rule first pre-emptively adds B to the right hand side of the sequent. The rule effectively uses the reverse persistence property: if some successor v forces A and rejects B, then the current world w must reject B too. These rules are very useful in our termination proof and saturation strategy in Section 3.4.

Contrary to `GHPC` and other traditional sequent calculi, our $\to_L$ rule and the symmetric $\prec_R$ contain implicit contractions on formulae other than just the principal formula. That is, during backward search, they carry their principal formula and all side formulae into the premises. Our rules $\wedge_L$, $\wedge_R$, $\vee_L$ and $\vee_R$ also carry their principal formula into their premises. We chose this approach because it allows us to give a semantic interpretation to the anti-axiom Ret. Because the static rules keep the principal formula from conclusion to premises, we can immediately deduce that a strongly saturated sequent, i.e., an instance of Ret, has a counter-model. The proof of Lemma 3.3.4, case "$\not\Rightarrow$", makes use of this property of our calculus.

Many of our rules use the generic "$\overset{?}{\Rightarrow}$" turnstile, and a clause that specifies whether the conclusion should have the "$\Rightarrow$" or the "$\not\Rightarrow$" turnstile. This indicates that various combinations of "$\Rightarrow$" and "$\not\Rightarrow$" are possible for the premises, and determines the turnstile of the conclusion in each case, as illustrated by the following example.

Example 3.2.6. *All of the following are possible instances of the* $\wedge_R$ *rule:*

1. *An instance which combines two derivations into a derivation:*

$$\frac{\emptyset\ q,r \ \Rightarrow\ q\wedge r, q\ \emptyset \qquad \emptyset\ q,r\ \Rightarrow\ q\wedge r, r\ \emptyset}{\emptyset\ q,r\ \Rightarrow\ q\wedge r\ \emptyset}\ \wedge_R$$

2. *An instance which combines a derivation and a refutation into a refutation:*

$$\frac{\emptyset\ q,r\ \Rightarrow\ q\wedge p, q\ \emptyset \qquad \{\{q,r\}\}\ q,r\ \not\Rightarrow\ q\wedge p, p\ \{\{q\wedge p, p\}\}}{\{\{q,r\}\}\ q,r\ \not\Rightarrow\ q\wedge p\ \{\{q\wedge p,p\}\}}\ \wedge_R$$

3. *An instance which combines a refutation and a derivation into a refutation:*

$$\frac{\{\{q,r\}\}\ q,r\ \not\Rightarrow\ p\wedge q, p\ \{\{p\wedge q, p\}\} \qquad \emptyset\ q,r\ \Rightarrow\ p\wedge q, q\ \emptyset}{\{\{q,r\}\}\ q,r\ \not\Rightarrow\ p\wedge q\ \{\{p\wedge q,p\}\}}\ \wedge_R$$

4. *An instance which combines two refutations into a refutation:*

$$\frac{\{\{t,r\}\}\ t,r\ \not\Rightarrow\ q\wedge p, q\ \{\{q\wedge p, q\}\} \qquad \{\{t,r\}\}\ t,r\ \not\Rightarrow\ q\wedge p, p\ \{\{q\wedge p, p\}\}}{\{\{t,r\}\}\ t,r\ \not\Rightarrow\ q\wedge p\ \{\{q\wedge p, q\}, \{q\wedge p, p\}\}}\ \wedge_R$$

As Example 3.2.6 shows, the conclusion of each of our rules *assigns the variables* based on the variables returned from the premise(s). In defining the rules, we use the indices $i, 1, 2$ to indicate the premise from which the variable takes its value. For rules with a single premise, the variables are simply passed down from premise to conclusion. For example, the conclusion of $\wedge_L$ has the same value of the variable $\mathcal{S}$ as the premise. However, for rules with multiple premises, we take a union of the sets of sets corresponding to each premise. For example, in Example 3.2.6(4) above, the $\mathcal{P}$ variable contains both $\{q\wedge p, q\}$ and $\{q\wedge p, p\}$, where the first set is from the left premise and the second set is from the right premise.

Thus the sets of sets stored in our variables *determinise* the return of formulae to lower sequents: semantically, each refutable premise corresponds to an open branch, and at this point we do not know whether it will stay open once processed in conjunction with lower sequents. Therefore, we need to temporarily keep all open branches. See also Remark 3.3.10 for a syntactic motivation for the set-of-sets concept.

The following are the *transitional rules* of our sequent calculus:

$$\frac{\mathcal{S}\ A \ \Rightarrow\ \Delta, B\ \mathcal{P}}{\mathcal{S}\ \Gamma, A \prec B \ \Rightarrow\ \Delta\ \mathcal{P}} \prec_{L1} \qquad \frac{\mathcal{S}\ \Gamma, A \ \Rightarrow\ B\ \mathcal{P}}{\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta, A \to B\ \mathcal{P}} \to_{R1}$$

$$\frac{\text{Prem}_1^{\prec} \quad \cdots \quad \text{Prem}_m^{\prec} \qquad \text{Prem}_1^{\to} \quad \cdots \quad \text{Prem}_n^{\to}}{\{\Gamma'\}\ \Gamma, A_1 \prec B_1, \cdots, A_m \prec B_m \ \not\Rightarrow\ \Delta, C_1 \to D_1, \cdots, C_n \to D_n\ \{\Delta'\}} \text{Refute}$$

where

(1) $\Gamma' = \Gamma, A_1 \prec B_1, \cdots, A_m \prec B_m$ (2) $\Delta' = \Delta, C_1 \to D_1, \cdots, C_n \to D_n$

(3) $\Gamma' \not\Rightarrow \Delta'$ is saturated

(4) Γ does not contain $\prec$-formulae and Δ does not contain $\to$-formuale

(5) $\forall i \in \{1, \cdots, m\}\ \forall j \in \{1, \cdots, n\}$:

(a) $\text{Prem}_i^{\prec} = \mathcal{S}_i^{\prec}\ A_i \ \not\Rightarrow\ \Delta', B_i\ \mathcal{P}_i^{\prec}$ (b) $\text{Prem}_j^{\to} = \mathcal{S}_j^{\to}\ \Gamma', C_j \ \not\Rightarrow\ D_j\ \mathcal{P}_j^{\to}$

(c) $\exists \Sigma \in \mathcal{S}_i^{\prec}.\Sigma \subseteq \Gamma'$ (d) $\exists \Pi \in \mathcal{P}_j^{\to}.\Pi \subseteq \Delta'$.

The $\to_{R1}$ rule is from Dragalin's `GHPC` [34], and the $\prec_{L1}$ is symmetric for the `DualInt` case: these rules introduce their principal $\to$-formula on the right or $\prec$-formula on the left. The Refute rule composes refutations of its premises to give a refutation of a sequent that may contain a number of $\to$-formulae on the right and $\prec$-formulae on the left. That the premises be refutable is stipulated by side conditions (5a) and (5b), which state that all premises have the "$\not\Rightarrow$" turnstile and hence are refutations. The extra side conditions (3), (4), (5c) and (5d) ensure that the conclusion of an instance of Refute is falsifiable (see the proof of Lemma 3.3.7) meaning that only certain refutations can be combined using this rule.

Example 3.2.7. *The following is an example instance of* Refute*:*

$$\frac{\{s\}\ s \ \not\Rightarrow\ b, a \to b, t\ \{b, a \to b, t\} \qquad \{r, s, q \to r, s \prec t, a\}\ r, s, q \to r, s \prec t, a \ \not\Rightarrow\ b\ \{b\}}{\{r, s, q \to r, s \prec t\}\ r, s, q \to r, s \prec t \ \not\Rightarrow\ b, a \to b\ \{b, a \to b\}} \text{Refute}$$

In this case:

- $m = 1, n = 1, \Gamma = \{r, s, q \to r\}, \Delta = \{b\}$
- *Prem*$_1^{\prec} = \{s\}\ s \ \not\Rightarrow\ b, a \to b, t\ \{b, a \to b, t\}$
- *Prem*$_1^{\to} = \{r, s, q \to r, s \prec t, a\}\ r, s, q \to r, s \prec t, a \ \not\Rightarrow\ b\ \{b\}$.

The following *special logical rules* are used to derive additional transitional rules:

$$\frac{\mathcal{S}_1\ \Gamma, \Pi_1 \ \Rightarrow\ \Delta\ \mathcal{P}_1 \quad \cdots \quad \mathcal{S}_m\ \Gamma, \Pi_m \ \Rightarrow\ \Delta\ \mathcal{P}_m}{\bigcup_1^m \mathcal{S}_i\ \Gamma, \bigvee(\{\Pi_1, \cdots, \Pi_m\}) \ \Rightarrow\ \Delta\ \bigcup_1^m \mathcal{P}_i} \bigvee_L$$

$$\frac{\mathcal{S}_1\ \Gamma \ \Rightarrow\ \Sigma_1, \Delta\ \mathcal{P}_1 \quad \cdots \quad \mathcal{S}_n\ \Gamma \ \Rightarrow\ \Sigma_n, \Delta\ \mathcal{P}_n}{\bigcup_1^n \mathcal{S}_i\ \Gamma \ \Rightarrow\ \bigwedge(\{\Sigma_1, \cdots, \Sigma_n\}), \Delta\ \bigcup_1^n \mathcal{P}_i} \bigwedge_R$$

$$\frac{\mathcal{S}_1\ \Gamma, A_1 \ \Rightarrow\ \Delta\ \mathcal{P}_1 \quad \cdots \quad \mathcal{S}_k\ \Gamma, A_k \ \Rightarrow\ \Delta\ \mathcal{P}_k}{\bigcup_1^k \mathcal{S}_i\ \Gamma, \bigwedge(\{\{A_1, \cdots, A_n\}\}) \ \Rightarrow\ \Delta\ \bigcup_1^k \mathcal{P}_i} \bigwedge_L \qquad \frac{\mathcal{S}\ \Gamma, \bigwedge \Sigma_1, \bigwedge \Sigma_2 \ \Rightarrow\ \Delta\ \mathcal{P}}{\mathcal{S}\ \Gamma, \bigwedge(\Sigma_1 \cup \Sigma_2) \ \Rightarrow\ \Delta\ \mathcal{P}} \bigwedge_L^{\cup}$$

$$\frac{\mathcal{S}_1\ \Gamma \ \Rightarrow\ A_1, \Delta\ \mathcal{P}_1 \quad \cdots \quad \mathcal{S}_k\ \Gamma \ \Rightarrow\ A_k, \Delta\ \mathcal{P}_k}{\bigcup_1^k \mathcal{S}_i\ \Gamma \ \Rightarrow\ \bigvee(\{\{A_1, \cdots, A_k\}\}), \Delta\ \bigcup_1^k \mathcal{P}_i} \bigvee_R \qquad \frac{\mathcal{S}\ \Gamma \ \Rightarrow\ \bigvee \Pi_1, \bigvee \Pi_2, \Delta\ \mathcal{P}}{\mathcal{S}\ \Gamma \ \Rightarrow\ \bigvee(\Pi_1 \cup \Pi_2), \Delta\ \mathcal{P}} \bigvee_R^{\cup}$$

These rules simply allow us to introduce extended `BiInt` formulae. The $\bigvee_L$ rule allows us to introduce $\bigvee$-formulae on the left, and the symmetric $\bigwedge_R$ rule allows us to introduce $\bigwedge$-formulae on the right. The $\bigwedge_L$ allows us to introduce a $\bigwedge$-formula, containing a single set containing a set of formulae, on the left, and the $\bigwedge_L^{\cup}$ rule allows us to introduce a larger $\bigwedge$-formula from two smaller ones; the $\bigvee_R$ and $\bigvee_R^{\cup}$ rules are dual.

The following *structural rules* are also used to derive additional transitional rules:

$$\frac{\mathcal{S}_1\ \Gamma \Rightarrow \Delta, A\ \mathcal{P}_1 \qquad \mathcal{S}_2\ A, \Gamma \Rightarrow \Delta\ \mathcal{P}_2}{\mathcal{S}_1 \cup \mathcal{S}_2\ \Gamma \Rightarrow \Delta\ \mathcal{P}_1 \cup \mathcal{P}_2}\ cut$$

$$\frac{\mathcal{S}\ \Gamma \Rightarrow \Delta\ \mathcal{P}}{\mathcal{S}\ A, \Gamma \Rightarrow \Delta\ \mathcal{P}}\ (\mathrm{LW}) \qquad \frac{\mathcal{S}\ \Gamma \Rightarrow \Delta\ \mathcal{P}}{\mathcal{S}\ \Gamma \Rightarrow \Delta, A\ \mathcal{P}}\ (\mathrm{RW})$$

Finally, the following *derived transitional rules* are used to achieve cut-free completeness, and their derived status will be explained shortly:

$$\frac{\mathcal{S}\ A \not\Rightarrow \Delta, B\ \mathcal{P} \qquad \mathcal{S}_1\ \Gamma, A \prec B, \Sigma_1 \overset{?}{\Rightarrow}_1 \Delta\ \mathcal{P}_1 \ \cdots\ \mathcal{S}_n\ \Gamma, A \prec B, \Sigma_n\ \Delta \overset{?}{\Rightarrow}_n \mathcal{P}_n}{\bigcup_1^n \mathcal{S}_i\ \Gamma, A \prec B \overset{?}{\Rightarrow}_0 \Delta\ \bigcup_1^n \mathcal{P}_i}\ \prec_{L2}$$

$$\text{Where } \mathcal{S} = \{\Sigma_1, \cdots, \Sigma_n\} \text{ for } n \geq 1 \text{ and } \overset{?}{\Rightarrow}_0 = \begin{cases} \Rightarrow & \text{if } \overset{?}{\Rightarrow}_i = \Rightarrow \text{ for all } 1 \leq i \leq n \\ \not\Rightarrow & \text{otherwise} \end{cases}$$

$$\frac{\mathcal{S}\ \Gamma, A \not\Rightarrow B\ \mathcal{P} \qquad \mathcal{S}_1\ \Gamma \overset{?}{\Rightarrow}_1 \Pi_1, \Delta, A \to B\ \mathcal{P}_1 \ \cdots\ \mathcal{S}_m\ \Gamma \overset{?}{\Rightarrow}_m \Pi_m, \Delta, A \to B\ \mathcal{P}_m}{\bigcup_1^m \mathcal{S}_i\ \Gamma \overset{?}{\Rightarrow}_0 \Delta, A \to B\ \bigcup_1^m \mathcal{P}_i}\ \to_{R2}$$

$$\text{Where } \mathcal{P} = \{\Pi_1, \cdots, \Pi_m\} \text{ for } m \geq 1 \text{ and } \overset{?}{\Rightarrow}_0 = \begin{cases} \Rightarrow & \text{if } \overset{?}{\Rightarrow}_i = \Rightarrow \text{ for all } 1 \leq i \leq m \\ \not\Rightarrow & \text{otherwise} \end{cases}$$

These rules compose a refutation of the *left-most premise* with one or more derivations/refutations of the *right premises*, where the formula-parts of the right premises contain formula sets like Σ_i and Π_i found in the variables of the left-most premise. That is, the right premise $\mathcal{S}_i\ \Gamma \overset{?}{\Rightarrow}_i \Pi_i, \Delta, A \to B\ \mathcal{P}_i$ of the $\to_{R2}$ rule contains the formula set $\Pi_i \in \mathcal{P}$, where $\mathcal{P}$ is one of the variables of the left-most premise.

We now explain how we can use the $\to_{R2}$ and $\prec_{L2}$ rules during backward search by giving an operational left-to-right reading for the rules. We first refute the left-most premise, which gives an instantiation of $\mathcal{S}$ and $\mathcal{P}$. In the $\to_{R2}$ case, we then extract the variable $\mathcal{P}$, and create $m \geq 1$ right premises, where each right premise corresponds to the conclusion together with additional formulae found in one of the members of $\mathcal{P}$. We then attempt to derive/refute the right premises using backward search, and put $\overset{?}{\Rightarrow}_0$ equal to "$\Rightarrow$" or "$\not\Rightarrow$" depending on whether or not all the right premises are derivable.

Having introduced all the sequent rules, we now define several sub-calculi that we shall use throughout the rest of the chapter: see Figure 3.1. **GBiInt**0 is the base system, which is sound (Lemma 3.3.8) and complete (although we do not show it), but uses

	GBiInt0	**GBiInt**1	**GBiInt**
Axioms, anti-axioms, static and transitional rules	✓	✓	✓
Special logical rules	✓	✓	
Structural rules	✓	✓	
Derived transitional rules		✓	✓

Figure 3.1: Calculi **GBiInt**0, **GBiInt**1 and **GBiInt**

the cut rule. **GBiInt**1 is obtained from **GBiInt**0 by adding two rules $\rightarrow_{R2}$ and $-\!\!<_{L2}$, which are **GBiInt**0-derivable and hence sound (Lemma 3.3.11). **GBiInt** is obtained from **GBiInt**1 by removing the special rules and structural rules and is cut-free, sound (Theorem 3.3.14) and complete (Theorem 3.4.15). **GBiInt** with additional blocking conditions is also the sequent calculus we use for backward search in Section 3.4. We use the generic name **GBiInt•** when we refer to any of the calculi, for example, in definitions, descriptions of rules and so on. Note that all these calculi are equivalent in terms of provability, and we only use separate calculi to structure soundness and completeness proofs.

Note that our main calculus **GBiInt** is cut-free: the cut rule is used only for showing the soundness of our derived transitional rules $\rightarrow_{R2}$ and $-\!\!<_{L2}$. Intuitively, we show how variable-passing absorbs essential instances of (cut) in a demand-driven way. Proving that variables absorb *all* essential cuts would give syntactic cut-admissibility.

GBiInt also has the subformula property. This is obvious for the LHS- and RHS-components of the sequents. For the variables, the subformula property is of a global nature: when the variables are instantiated at instances of the Ret anti-axiom, they take values from the LHS- and RHS-components of this anti-axiom. When the variables are passed down towards the root of refutations, they are combined using the union operator, so no new formulae are created. Thus all formulae are subformulae of the end-sequent.

GBiInt is also free of all other structural rules; that is, explicit contraction and weakening is not required to achieve completeness. We could have also started with sequents as multisets instead of sets and shown that contraction is admissible, but since all our static rules contain implicit contractions, this would be a simple and redundant exercise.

Definition 3.2.8. *A* **GBiInt•** *tree is a tree of sequents where each leaf is an instance of the* **GBiInt•** *axioms or anti-axiom, and parents are obtained from children by instantiating a* **GBiInt•** *rule. The height of a* **GBiInt•** *tree is the number of sequents on the longest branch. A* derivation *is a* **GBiInt•** *tree rooted at* $\mathcal{S}\ \Gamma\ \Rightarrow\ \Delta\ \mathcal{P}$*. A sequent is* derivable *if there exists a derivation for it; we write* $\vdash\ \mathcal{S}\ \Gamma\ \Rightarrow\ \Delta\ \mathcal{P}$*. A* refutation *is a* **GBiInt•** *tree rooted at* $\mathcal{S}\ \Gamma\ \not\Rightarrow\ \Delta\ \mathcal{P}$*. A sequent is* refutable *if there exists a refutation for it; we write* $\vdash \mathcal{S}\ \Gamma\ \not\Rightarrow\ \Delta\ \mathcal{P}$*.*

We deliberately use $\vdash$ for both derivability and refutability to emphasise their first-class status.

3.2.3 Example

We now revisit Uustalu's example [95] that we first saw in Section 2.2.4, and show the full derivation of this example using **GBiInt**. In all cases below, $X := (p \mathrel{-\!\!<} q) \wedge r$.
Let (1) be the refutation below:

$$\dfrac{\dfrac{}{\{\{p,r,q\}\}\ p,r,q \ \not\Rightarrow\ X,\mathbf{p}\prec\mathbf{q}\ \{\{X,\mathbf{p}\prec\mathbf{q}\}\}}\,\text{Ret}\qquad \dfrac{}{\emptyset\ p,r \ \Rightarrow\ X,p\prec q,p\ \emptyset}\,id}{\{\{p,r,q\}\}\ p,r \ \not\Rightarrow\ X,p\prec q\ \{\{X,\mathbf{p}\prec\mathbf{q}\}\}}\,\prec_R$$

Let (2) be the derivation below:

$$\dfrac{\dfrac{}{\emptyset\ p,\mathbf{q} \ \Rightarrow\ q,r\to X,X,p\prec q\ \emptyset}\,id\qquad \dfrac{}{\emptyset\ p \ \Rightarrow\ q,r\to X,X,p\prec q,\mathbf{p}\ \emptyset}\,id}{\emptyset\ p \ \Rightarrow\ q,r\to X,X,\mathbf{p}\prec\mathbf{q}\ \emptyset}\,\prec_R$$

Then the following is a cut-free derivation of Uustalu's [95] formula $p \to (q \vee (r \to ((p \mathrel{-\!\!<} q) \wedge r))$, simplified to the sequent $p \overset{?}{\Rightarrow} q, r \to ((p \mathrel{-\!\!<} q) \wedge r)$:

$$\dfrac{\dfrac{(1)\qquad \dfrac{}{\emptyset\ p,r \ \Rightarrow\ X,r\ \emptyset}\,id}{\{\{p,r,q\}\}\ p,r \ \not\Rightarrow\ (p\prec q)\wedge r\ \{\{X,\mathbf{p}\prec\mathbf{q}\}\}}\,\wedge_R\qquad (2)}{\emptyset\ p \ \Rightarrow\ q,r\to((p\prec q)\wedge r)\ \emptyset}\,\to_{R2}$$

The top left anti-axiom in (1) is an instance of Ret because the sequent is strongly saturated. The variables $\mathcal{S}$ and $\mathcal{P}$ that are assigned at this Ret anti-axiom transmit information down to the parents and across to their siblings via the $\to_{R2}$ rule.

The key to the derivation is the bolded $\mathbf{p} \mathrel{-\!\!<} \mathbf{q}$ formula that occurs in the variable $\mathcal{P}$ of the left-most leaf of (1) and in the RHS of the right premise (2) of $\to_{R2}$. Note that $\to_{R2}$ has only one right premise here, since the $\mathcal{P}$ variable contains only one set of formulae.

We can also read the above derivation as a backward search. We start with the end-sequent $p \overset{?}{\Rightarrow} q, r \to ((p \prec q) \wedge r)$, which we want to prove or refute. Since the only possible principal formula is a $\to$-formula on the right, we know that we need to use either $\to_{R1}$, $\to_{R2}$ or Refute. In all cases, we need to consider the sequent $p, r \overset{?}{\Rightarrow} (p \prec q) \wedge r$. We then find a refutation of the sequent $p, r \overset{?}{\Rightarrow} (p \prec q) \wedge r$, obtaining $\{\{p,r,q\}\}\ p,r \not\Rightarrow (p \prec q) \wedge r\ \{\{X, \mathbf{p} \prec \mathbf{q}\}\}$ and thus receiving back the variables $\mathcal{S} = \{\{p,r,q\}\}$ and $\mathcal{P} = \{\{X, \mathbf{p} \prec \mathbf{q}\}\}$. We then apply the $\to_{R2}$ rule since its side conditions are met. The left premise is $\{\{p,r,q\}\}\ p,r \not\Rightarrow (p \prec q) \wedge r\ \{\{X, \mathbf{p} \prec \mathbf{q}\}\}$, and we create a single right premise because the $\mathcal{P}$ variable contains a single member $\{X, \mathbf{p} \prec \mathbf{q}\}$. Since the right premise is derivable, so is the end-sequent, so we put $\overset{?}{\Rightarrow} = \Rightarrow$ and obtain $\emptyset\ p \Rightarrow q, r \to ((p \prec q) \wedge r)\ \emptyset$.

3.3 Soundness and completeness

In this section, we prove the soundness and completeness of **GBiInt** with respect to the semantics of `BiInt` (recall Section 2.2.4.4 where we introduced `BiInt` semantics). We start by proving that the base rules of **GBiInt**0 are sound.

3.3.1 Soundness of GBiInt0

We first observe that the variables are empty at the root of derivations.

Lemma 3.3.1. *If $\mathcal{S}\ \Gamma\ \Rightarrow\ \Delta\ \mathcal{P}$ is derivable then $\mathcal{S} = \emptyset$ and $\mathcal{P} = \emptyset$.*

Proof. By induction on the height of the given derivation. This is obvious for all **GBiInt•** rules which combine "$\Rightarrow$"-premises into a "$\Rightarrow$"-conclusion since the union of empty sets is $\emptyset$.

The more interesting cases are the rules $\rightarrow_{R2}$ and $-\!\!<_{L2}$, which combine a refutation ("$\not\Rightarrow$") of the left-most premise with a combination of refutations ("$\not\Rightarrow$") or derivations ("$\Rightarrow$") of the right premises. Here, the variables at the conclusion are the union of the variables of the right premises only. Since the condition in these rules specifies that a derivation of the conclusion is obtained only when *all* the *right* premises are derivations, again the variables at the conclusion are the union of empty sets, giving the empty set as required. Q.E.D.

We will prove soundness of the rules by showing that each rule preserves the semantic judgement σ from (3.2.4) and (3.2.5) downwards, so we start by formally defining this concept.

Definition 3.3.2. *A* **GBiInt•** *rule ρ with conclusion $\mathcal{S}_0\ \Gamma_0\ \overset{?}{\Rightarrow}_0\ \Delta_0\ \mathcal{P}_0$ and $n \geq 1$ premises, with i-th premise $\mathcal{S}_i\ \Gamma_i\ \overset{?}{\Rightarrow}_i\ \Delta_i\ \mathcal{P}_i$, preserves the semantic judgement σ downwards if every premise is the conclusion of a* **GBiInt•** *derivation or refutation and:*

$$\forall i \in \{1, \cdots, n\}.\ \textit{if}\ \ \sigma(\mathcal{S}_i\ \Gamma_i\ \overset{?}{\Rightarrow}_i\ \Delta_i\ \mathcal{P}_i)\ \textit{then}\ \sigma(\mathcal{S}_0\ \Gamma_0\ \overset{?}{\Rightarrow}_0\ \Delta_0\ \mathcal{P}_0).$$

Lemma 3.3.3. *The static, special, $\rightarrow_{R1}$ and $-\!\!<_{L1}$ logical rules, and all structural rules preserve the semantic judgement σ downwards.*

Proof. Easily follows from translations 3.2.4 and 3.2.5 and the definitions of the rules. Q.E.D.

Lemma 3.3.4. *The semantic judgement σ holds at the leaves of* **GBiInt•** *trees. That is, the $\Rightarrow$-leaves are valid, and the $\not\Rightarrow$-leaves are falsifiable.*

Proof.

$\Rightarrow$: A leaf $\Gamma \Rightarrow \Delta$ must be an instance of *id*, $\bot_L$ or $\top_R$. In all cases, the corresponding formula shown below is valid:

$$\begin{array}{lllllll} id & \bigvee\emptyset \wedge \hat{\Gamma} \wedge A & \rightarrow & \check{\Delta} \vee A \vee \bigwedge\emptyset & = & \hat{\Gamma} \wedge A & \rightarrow & \check{\Delta} \vee A \\ \bot_L & \bigvee\emptyset \wedge \hat{\Gamma} \wedge \bot & \rightarrow & \check{\Delta} \vee \bigwedge\emptyset & = & \hat{\Gamma} \wedge \bot & \rightarrow & \check{\Delta} \\ \top_R & \bigvee\emptyset \wedge \hat{\Gamma} & \rightarrow & \check{\Delta} \vee \top \vee \bigwedge\emptyset & = & \hat{\Gamma} & \rightarrow & \check{\Delta} \vee \top. \end{array}$$

$\not\Rightarrow$: A leaf $\Gamma \not\Rightarrow \Delta$ must be an instance of Ret. We show that the corresponding formula $\bigvee\mathcal{S} \wedge \hat{\Gamma} \rightarrow \check{\Delta} \vee \bigwedge\mathcal{P}$ is falsifiable. Since Ret assigns $\mathcal{S} := \{\Gamma\}$ and $\mathcal{P} := \{\Delta\}$, the corresponding formula under translation τ is

$$\bigvee\{\Gamma\} \wedge \hat{\Gamma} \to \check{\Delta} \vee \bigwedge\{\Delta\} \quad = \quad \hat{\Gamma} \wedge \hat{\Gamma} \to \check{\Delta} \vee \check{\Delta} \quad = \quad \hat{\Gamma} \to \check{\Delta}.$$

To falsify $\hat{\Gamma} \to \check{\Delta}$, we create a model with a single reflexive world w_0, and for every atom p in Γ, we let $V(p) = \{w_0\}$, and for every atom q in Δ, we let $V(q) = \emptyset$. An atom cannot be both in Γ and Δ since $\Gamma \not\Rightarrow \Delta$ must be strongly saturated and thus $\Gamma \cap \Delta = \emptyset$.

To show that $\hat{\Gamma} \to \check{\Delta}$ is falsifiable at w_0, we need to show that $w_0 \Vdash \Gamma$ and $w_0 \dashv\mid \Delta$. For every atom in Γ and Δ, the valuation ensures both. For every composite formula A, we do a simultaneous induction on its length. Since the side condition of Ret implies that $\Gamma \not\Rightarrow \Delta$ is strongly saturated, we know that the required subformulae are already in Γ or Δ as appropriate, and they fall under the induction hypothesis.

Thus we know that $w_0 \Vdash \Gamma$ and $w_0 \dashv\mid \Delta$, therefore $\hat{\Gamma} \to \check{\Delta}$ is falsifiable.

Q.E.D.

Definition 3.3.5. *Given an instance of* Refute, *we use* $\to$*-premises to refer to the premises* $\text{Prem}_1^{\to}, \cdots, \text{Prem}_n^{\to}$ *, and* $-\!\!<$*-premises to refer to the premises* $\text{Prem}_1^{-<}, \cdots, \text{Prem}_m^{-<}$.

Definition 3.3.6. *Given two valuations* $\vartheta_1 = Atoms_1 \cup \{\top, \bot\} \to 2^{W_1}$ *and* $\vartheta_2 = Atoms_2 \cup \{\top, \bot\} \to 2^{W_2}$ *with* $W_1 \cap W_2 = \emptyset$*, we define the disjoint union of* ϑ_1 *and* ϑ_2 *as a set of pairs:*

$$\vartheta_1 \cup \vartheta_2 := \{(p, S) \quad | \quad p \in Atoms_1 \cup Atoms_2 \cup \{\top, \bot\} \text{ and } S = \{w \mid w \in W_1 \cup W_2 \text{ and } [w \in \vartheta_1(p) \text{ or } w \in \vartheta_2(p)]\}.$$

The proof of the next lemma has similarities to parts of a traditional completeness proof.

Lemma 3.3.7. *The* Refute *rule preserves the semantic judgement downwards.*

Proof. We assume that the semantic judgement σ holds for all the premises, and show that σ holds for the conclusion. That is, we assume that all the premises are falsifiable and show that the conclusion is falsifiable. To show that the conclusion is falsifiable, we need to show that there exists a `BiInt` model $M = \langle W, \leq, V\rangle$ and a world $w \in W$ such that $w \Vdash \Gamma'$ and $w \dashv\mid \Delta'$. We construct the model as follows:

Step 1. Let $W := \{w_0\}$ and $\leq_0 := \{(w_0, w_0)\}$.

Step 2. For all atoms $p \in \Gamma$, let $V(p) := \{w_0\}$. For all atoms $q \in \Delta$, let $V(q) := \emptyset$.
That is, the valuation makes every atom in Γ true at w_0. Since side condition (3) of Refute ensures that the conclusion is saturated, Definition 3.2.4 implies $\Gamma \cap \Delta = \emptyset$, and hence the valuation makes every atom in Δ false at w_0. Then, since the conclusion is saturated, induction on the size of members of Γ and Δ gives $w_0 \Vdash \Gamma$ and $w_0 \dashv\mid \Delta$.

Step 3. For each $A_i -\!\!< B_i \in \Gamma'$:

(a) Since the premise $Prem_i^{\prec} = \mathcal{S}_i^{\prec}\ A_i \ \not\Rightarrow\ \Delta', B_i\ \mathcal{P}_i^{\prec}$ is falsifiable by assumption, we know there exists a `BiInt` model $M_i = \langle W_i, \leq_i, \vartheta_i \rangle$ and a world $w_i \in W_i$ such that $w_i \Vdash \bigvee \mathcal{S}_i^{\prec}, A_i$ and $w_i \dashv\Vert \Delta', B_i, \bigwedge \mathcal{P}_i^{\prec}$. If necessary, we rename the worlds in W_i to ensure their names are disjoint from the names of worlds already in W.

(b) Let $W := W \cup W_i$.

(c) Let $\leq_0 := \leq_0 \cup \leq_i \cup \{(w_i, w_0)\}$ thus making w_i an $\leq_0$-predecessor of w_0.

(d) Let $V := V \cup \vartheta_i$ using Definition 3.3.6.

Step 4. For each $C_j \to D_j \in \Delta'$, perform an analogous procedure to Step 3, using $Prem_j^{\to} = \mathcal{S}_j^{\to}\ \Gamma', C_j \ \not\Rightarrow\ D_j\ \mathcal{P}_j^{\to}$, except sub-step (c) becomes $\leq_0 := \leq_0 \cup \leq_j \cup \{(w_0, w_j)\}$.

Step 5. Let $\leq$ be the transitive closure of $\leq_0$.

Step 6. We now have that $\langle W, \leq \rangle$ is a `BiInt` frame.

Step 7. To show that $M = \langle W, \leq, V \rangle$ is a `BiInt` model, we also need to show that it obeys persistence. From Steps 3 and 4 we know that w_0 has $\leq_0$-predecessors w_i and $\leq_0$-successors w_j. Forward persistence holds between all w_i and w_0, and between w_0 and all w_j because:

(a) Step 3 gives (i) $w_i \Vdash \bigvee \mathcal{S}_i^{\prec}, A_i$. We have $A_i \in \Gamma'$ because $A_i {-\!\!<} B_i \in \Gamma'$ and condition (3) of Refute implies $\Gamma' \not\Rightarrow \Delta'$ is saturated. Therefore (ii) $w_0 \Vdash A_i$. We have (iii) $w_0 \Vdash \bigvee \mathcal{S}_i^{\prec}$ because of side condition (5c) of Refute and the semantics of the $\bigvee$ connective: see Definition 3.2.2. From (ii) and (iii) we get $w_0 \Vdash \bigvee \mathcal{S}_i^{\prec}, A_i$, and so every formula forced by the i-th $\leq_0$-predecessor w_i is also forced by w_0.

(b) By inspection, all $\to$-premises 1 to n contain Γ'. This gives us that every formula found in Γ' and hence forced by w_0 is also forced by all $\leq_0$-successors in Step 4.

Similarly, reverse persistence holds because of side condition (5d) of Refute and the fact that all ${-\!\!<}$-premises 1 to m contain Δ'.

To show that persistence holds for all $\leq$-related worlds, we use transitivity of the subset relation and the initial assumption, specifically the fact that persistence holds in all models M_i and M_j used in Steps 3 and 4.

Q.E.D.

Lemma 3.3.8. **GBiInt**0 *is sound.*

Proof. By Lemmas 3.3.3 to 3.3.7. Q.E.D.

3.3.2 Soundness of **GBiInt**1

The only difference between **GBiInt**0 and **GBiInt**1 is that **GBiInt**1 contains the extra transitional rules $\rightarrow_{R2}$ and $\prec_{L2}$. We now show that each of these rules is derivable in **GBiInt**0. In particular, instances of $\rightarrow_{R2}$ and $\prec_{L2}$ can be seen as absorbing certain instances of cut and weakening. Since **GBiInt**0 is sound, so are the extra derived rules.

The following lemma is crucial for showing the soundness of $\rightarrow_{R2}$ and $\prec_{L2}$ because it shows that the variables at the root of a refutation in fact contain the information required to turn the refutation into a derivation. More specifically, we will use the $\mathcal{S}$ variable to obtain a derivation from the refutation when we apply the $\prec_{L2}$ rule, and will use the $\mathcal{P}$ variable when we apply the $\rightarrow_{R2}$ rule. Reading **GBiInt**$\bullet$ trees top-down, we do not know which variable will be required at a lower sequent, so we keep both $\mathcal{S}$ and $\mathcal{P}$.

Lemma 3.3.9. *For all $\mathcal{S}, \Gamma, \Delta, \mathcal{P}$:*

$$\text{if} \vdash \mathcal{S}\ \Gamma \not\Rightarrow \Delta\ \mathcal{P}, \text{ then} \vdash \emptyset\ \textstyle\bigwedge \mathcal{P}, \Gamma \Rightarrow \Delta\ \emptyset \text{ and} \vdash \emptyset\ \Gamma \Rightarrow \Delta, \textstyle\bigvee \mathcal{S}\ \emptyset.$$

Proof. By induction on the height of the refutation of $\mathcal{S}\ \Gamma \not\Rightarrow \Delta\ \mathcal{P}$.
Base Case: A refutation of height 1 must be an instance of Ret:

$$\{\Gamma\}\ \Gamma \not\Rightarrow \Delta\ \{\Delta\}\ \text{Ret} \quad \text{where } \Gamma \not\Rightarrow \Delta \text{ is strongly saturated}$$

That is, $\mathcal{S} = \{\Gamma\}$ and hence $\emptyset\ \Gamma \Rightarrow \Delta, \bigvee \mathcal{S}\ \emptyset$ is $\emptyset\ \Gamma \Rightarrow \Delta, \bigvee\{\Gamma\}\ \emptyset$. Then the following is a derivation of $\emptyset\ \Gamma \Rightarrow \Delta, \bigvee\{\Gamma\}\ \emptyset$, where $\Gamma = \{\gamma_1, \cdots, \gamma_k\}$ for some $k \geq 1$:

$$\dfrac{\dfrac{}{\emptyset\ \Gamma \Rightarrow \Delta, \gamma_1\ \emptyset}\,id \quad \cdots \quad \dfrac{}{\emptyset\ \Gamma \Rightarrow \Delta, \gamma_k\ \emptyset}\,id}{\emptyset\ \Gamma \Rightarrow \Delta, \bigvee\{\Gamma\}\ \emptyset}\,\bigvee_R$$

Dually for $\bigwedge \mathcal{P}$ on the left.
IH: Assume the lemma holds for all refutations of height $\leq k$, and for all $\mathcal{S}, \Gamma, \Delta, \mathcal{P}$.
Induction step: Consider a refutation of height $k+1$, and the lowest rule application.

There are two cases:

Case 1: For all rules except Refute, we can use the induction hypothesis for the premises to easily obtain the required derivation. For example, consider the $\wedge_R$ rule:

$$\dfrac{\mathcal{S}_1\ \Gamma \overset{?}{\Rightarrow}_1 \Delta, A \wedge B, A\ \mathcal{P}_1 \qquad \mathcal{S}_2\ \Gamma \overset{?}{\Rightarrow}_2 \Delta, A \wedge B, B\ \mathcal{P}_2}{\mathcal{S}_1 \cup \mathcal{S}_2\ \Gamma \overset{?}{\Rightarrow}_0 \Delta, A \wedge B\ \mathcal{P}_1 \cup \mathcal{P}_2}\,\wedge_R$$

$$\text{Where } \overset{?}{\Rightarrow}_0 = \begin{cases} \Rightarrow & \text{if } \overset{?}{\Rightarrow}_1 = \Rightarrow \text{ and } \overset{?}{\Rightarrow}_2 = \Rightarrow \\ \not\Rightarrow & \text{otherwise} \end{cases}$$

Since the conclusion is refutable by assumption, we know that $\overset{?}{\Rightarrow}_0 = \not\Rightarrow$, then by the condition of the rule one or both of $\overset{?}{\Rightarrow}_1$ and $\overset{?}{\Rightarrow}_2$ is also $\not\Rightarrow$.

The cases when either $\overset{?}{\Rightarrow}_1 = \not\Rightarrow$ or $\overset{?}{\Rightarrow}_2 = \not\Rightarrow$ are straightforward. If both $\overset{?}{\Rightarrow}_1 = \not\Rightarrow$ and $\overset{?}{\Rightarrow}_2 = \not\Rightarrow$ then both premises are the roots of refutations of height $\leq k$. Then the induction hypothesis gives derivations δ_1 of $\emptyset\ \bigwedge \mathcal{P}_1, \Gamma \Rightarrow \Delta, A \wedge B, A\ \emptyset$ and δ_2 of $\emptyset\ \bigwedge \mathcal{P}_2, \Gamma \Rightarrow \Delta, A \wedge B, B\ \emptyset$, from which we obtain the following derivation:

$$\dfrac{\dfrac{\dfrac{\delta_1}{\emptyset\ \bigwedge\mathcal{P}_1, \Gamma\ \Rightarrow\ \Delta, A\wedge B, A\ \emptyset}}{\emptyset\ \bigwedge\mathcal{P}_1, \bigwedge\mathcal{P}_2, \Gamma\ \Rightarrow\ \Delta, A\wedge B, A\ \emptyset}\text{LW} \qquad \dfrac{\dfrac{\delta_2}{\emptyset\ \bigwedge\mathcal{P}_2, \Gamma\ \Rightarrow\ \Delta, A\wedge B, B\ \emptyset}}{\emptyset\ \bigwedge\mathcal{P}_1, \bigwedge\mathcal{P}_2, \Gamma\ \Rightarrow\ \Delta, A\wedge B, B\ \emptyset}\text{LW}}{\dfrac{\emptyset\ \bigwedge\mathcal{P}_1, \bigwedge\mathcal{P}_2, \Gamma\ \Rightarrow\ \Delta, A\wedge B\ \emptyset}{\emptyset\ \bigwedge(\mathcal{P}_1\cup\mathcal{P}_2), \Gamma\ \Rightarrow\ \Delta, A\wedge B\ \emptyset}\textstyle\bigwedge_L^{\cup}}\wedge_R$$

Dually for $\bigvee\mathcal{S}$ on the right.

Case 2: Consider the Refute rule. That is, $\mathcal{S} = \{\Gamma'\}$ and hence $\emptyset\ \Gamma'\ \Rightarrow\ \Delta', \bigvee\mathcal{S}\ \emptyset$ is $\emptyset\ \Gamma'\ \Rightarrow\ \Delta', \bigvee\{\Gamma'\}\ \emptyset$. Then a derivation of $\emptyset\ \Gamma'\ \Rightarrow\ \Delta', \bigvee\{\Gamma'\}\ \emptyset$, where $\Gamma' = \Gamma, A_1 \prec B_1, \cdots, A_n \prec B_n = \{\gamma_1, \cdots, \gamma_k\}$ for some $k \geq 1$ is:

$$\dfrac{\dfrac{}{\emptyset\ \Gamma'\ \Rightarrow\ \Delta', \gamma_1\ \emptyset}id \qquad \cdots \qquad \dfrac{}{\emptyset\ \Gamma'\ \Rightarrow\ \Delta', \gamma_k\ \emptyset}id}{\emptyset\ \Gamma'\ \Rightarrow\ \Delta', \bigvee\{\Gamma'\}\ \emptyset}\textstyle\bigvee_R$$

Dually for $\bigwedge\mathcal{P}$ on the left. Q.E.D.

Remark 3.3.10. *Case 1 of the induction step in the previous proof shows why we need to keep variables as sets of sets and form the union of variables from all premises. If we only kept, say $\mathcal{P}_1$, at the conclusion of the $\wedge_R$ rule, the above case would not go through since we would not be able to show that the right premise is derivable.*

We now show that the $\rightarrow_{R2}$ and $\prec_{L2}$ rules are sound by showing how they absorb certain instances of *cut*. Intuitively, we show that instead of guessing the cut formula required for a derivation, we can combine the variables at the root of the refutation of the left premise with a derivation of the right premise to obtain the derivation of the conclusion.

Lemma 3.3.11. *The $\rightarrow_{R2}$ and $\prec_{L2}$ rules are sound.*

Proof. We show the case for $\rightarrow_{R2}$, the case for $\prec_{L2}$ is symmetric.

$$\dfrac{\mathcal{S}\ \Gamma, A\ \not\Rightarrow\ B\ \mathcal{P} \qquad \mathcal{S}_1\ \Gamma\ \overset{?}{\Rightarrow}_1\ \Pi_1, \Delta, A\rightarrow B\ \mathcal{P}_1\ \cdots\ \mathcal{S}_m\ \Gamma\ \overset{?}{\Rightarrow}_m\ \Pi_m, \Delta, A\rightarrow B\ \mathcal{P}_m}{\bigcup_1^m \mathcal{S}_i\ \Gamma\ \overset{?}{\Rightarrow}_0\ \Delta, A\rightarrow B\ \bigcup_1^m \mathcal{P}_i}\rightarrow_{R2}$$

There are two cases: either all the right premises are derivable, or at least one is refutable.

1. If all the right premises are derivable, then $\overset{?}{\Rightarrow}_0 = \Rightarrow$, i.e. the conclusion is also derivable. We show how to replace an instance of $\rightarrow_{R2}$ with a sound instance of the cut rule.

   The left-most premise $\mathcal{S}\ \Gamma, A\ \not\Rightarrow\ B\ \mathcal{P}$ of $\rightarrow_{R2}$ is refutable. Then by Lemma 3.3.9, there is a derivation δ_1 of the sequent $\emptyset\ \bigwedge\mathcal{P}, \Gamma, A\ \Rightarrow\ B\ \emptyset$.

   All the right premises of $\rightarrow_{R2}$ are derivable, that is, $\mathcal{S}_i\ \Gamma\ \Rightarrow\ \Delta, A\rightarrow B, \Pi_i\ \mathcal{P}_i$ has a derivation δ_2^i, for $1 \leq i \leq n$ and $n \geq 1$. By Lemma 3.3.1, we have that $\mathcal{S}_i = \emptyset$ and $\mathcal{P}_i = \emptyset$, thus each δ_2^i is a derivation of $\emptyset\ \Gamma\ \Rightarrow\ \Delta, A\rightarrow B, \Pi_i\ \emptyset$. Then let δ_2 be a derivation of $\emptyset\ \Gamma\ \Rightarrow\ \Delta, A\rightarrow B, \bigwedge\mathcal{P}\ \emptyset$, where $\mathcal{P} = \{\Pi_1, \cdots, \Pi_n\}$ for $n \geq 1$, as shown below:

$$\dfrac{\begin{matrix}\delta_2^1 \\ \emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B, \Pi_1\ \emptyset\end{matrix} \qquad \cdots \qquad \begin{matrix}\delta_2^n \\ \emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B, \Pi_n\ \emptyset\end{matrix}}{\emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B, \bigwedge \mathcal{P}\ \emptyset}\ \bigwedge_R$$

Then a cut on $\bigwedge \mathcal{P}$ gives a sound derivation of the conclusion of $\to_{R2}$ as follows:

$$\dfrac{\begin{matrix}\delta_2 \\ \emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B, \bigwedge \mathcal{P}\ \emptyset\end{matrix} \qquad \dfrac{\begin{matrix}\delta_1 \\ \emptyset\ \bigwedge \mathcal{P}, \Gamma, A \ \Rightarrow\ B\ \emptyset\end{matrix}}{\emptyset\ \bigwedge \mathcal{P}, \Gamma \ \Rightarrow\ \Delta, A \to B\ \emptyset}\ \to_{R1}}{\emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B\ \emptyset}\ cut$$

Thus, if $\emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B\ \emptyset$ is the conclusion of $\to_{R2}$ in **GBiInt**1, then there is a **GBiInt**0 derivation of $\emptyset\ \Gamma \ \Rightarrow\ \Delta, A \to B\ \emptyset$. We have used only the part of Lemma 3.3.9 relating to $\mathcal{P}$. The symmetric case of $-\!\!<_{L2}$ requires the part relating to $\mathcal{S}$.

2. If any right premise is refutable, then $\overset{?}{\Rightarrow}_0 = \not\Rightarrow$, i.e. the conclusion is also refutable. But the RHS of each right premise contains the RHS of the conclusion, while the LHSs are the same, so if any right premise is falsifiable, then the conclusion is also falsifiable.

Q.E.D.

We now illustrate the effect of the transformation in the previous lemma by showing the example derivation of Section 3.2.3 using an instance of cut instead of an instance of $\to_{R2}$.

Example 3.3.12 (Derivation using cut)**.** *Below is a* **GBiInt**1*-derivation of Uustalu's [95] interaction formula* $p \to (q \vee (r \to ((p-\!\!<q) \wedge r))$*, simplified to the sequent* $p \overset{?}{\Rightarrow} q, r \to ((p-\!\!<q) \wedge r)$*. Let* $X := (p-\!\!<q) \wedge r$ *and* $Y = r \to X$*. This derivation uses a cut on* $p-\!\!<q$ *instead of* $\to_{R2}$ *and variables. All variables have a value of* $\emptyset$ *so we omit them to save space.*

$$\dfrac{\dfrac{\dfrac{}{q, p \Rightarrow q, Y, p \prec q}\ Id \qquad \dfrac{}{p \Rightarrow p, q, Y, p \prec q}\ Id}{p \Rightarrow q, Y, p \prec q}\ \prec_R \qquad \dfrac{\dfrac{\dfrac{}{p \prec q, p, r \Rightarrow X, p \prec q}\ Id \qquad \dfrac{}{p \prec q, p, r \Rightarrow X, r}\ Id}{p \prec q, p, r \Rightarrow (p \prec q) \wedge r}\ \wedge_R}{p \prec q, p \Rightarrow q, r \to ((p \prec q) \wedge r)}\ \to_{R1}}{p \Rightarrow q, r \to ((p \prec q) \wedge r)}\ cut$$

Comparing the derivation of Example 3.3.12 with that of Uustalu's original derivation (Example 2.2.4) shows that their basic structure is the same. There are some notational differences since **GBiInt**1 uses variables (which are empty and omitted in this case). Also, the cut rule in **GBiInt**1 is additive rather than multiplicative to ensure an easy transformation of instances of $\to_{R2}$ and $-\!\!<_{L2}$ into cut. A more significant difference is the hidden contraction in $\prec_R$ and $\wedge_R$, where the principal formula is carried from the conclusion to the premises. As a consequence, the axioms of the derivation of Example 3.3.12 contain additional formulae to those found in Example 2.2.4. The additional formulae, and hence the contractions, are redundant in this case since the rules have produced a derivation, but would be essential otherwise.

3.3.3 Soundness and completeness of GBiInt

The difference between **GBiInt**1 and **GBiInt** is that **GBiInt** omits the structural rules and the special rules for $\bigwedge$ and $\bigvee$. Thus **GBiInt** is sound, since it is a subset of **GBiInt**1, whose soundness we showed in the previous section. To show the completeness of **GBiInt**, we will use the fact that all the rules preserve the semantic judgement σ downwards, and that if a sequent is not derivable, then it is refutable. That is, a refutation gives a semantically correct counter-model, and we can obtain such a refutation whenever we cannot obtain a derivation. We now prove the precursor to the soundness and completeness corollaries.

Theorem 3.3.13. *For all $\mathcal{S}, \Gamma, \Delta, \mathcal{P}$:*

1. *If $\vdash \mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P}$ then $\models_{\texttt{BiInt}} \tau(\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P})$.*
2. *If $\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P}$ then $\dashv_{\texttt{BiInt}} \tau(\mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P})$.*

Proof. We proceed by simultaneous induction on the height of the derivation or refutation. If the height is 1, we have a leaf node, so both cases 1 and 2 follow by Lemma 3.3.4. For all $\mathcal{S}, \Gamma, \Delta, \mathcal{P}$, assume the lemma holds for all derivations/refutations of height $\leq k$. Let ρ be the lowest rule application of a derivation/refutation of height $k+1$. The premises of ρ obey the IH. By Lemmas 3.3.3, 3.3.11 and 3.3.7, ρ preserves the semantic judgement σ downwards. Thus, cases 1 and 2 hold for the conclusion of ρ. Q.E.D.

In some sense, Theorem 3.3.13 is our main result, since it shows that derivability/refutability captures validity/falsifiability. But traditionally, we wish to obtain soundness, which says that derivability implies validity, and completeness, which says that validity implies derivability. The traditional soundness result easily follows from Theorem 3.3.13, as shown below:

Corollary 3.3.14 (Soundness)**.** *If $\vdash \mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P}$ then $\models_{\texttt{BiInt}} \hat{\Gamma} \to \check{\Delta}$.*

Proof. By case 1 of Theorem 3.3.13, we have $\models_{\texttt{BiInt}} \tau(\mathcal{S}\ \Gamma \ \Rightarrow\ \Delta\ \mathcal{P})$. Then by Lemma 3.3.1, we have $\mathcal{S} = \emptyset = \mathcal{P}$ and thus $\models_{\texttt{BiInt}} \hat{\Gamma} \to \check{\Delta}$. Q.E.D.

Corollary 3.3.15 (Pre-completeness)**.** *If $\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P}$ then $\dashv_{\texttt{BiInt}} \hat{\Gamma} \to \check{\Delta}$.*

Proof. By case 2 of Theorem 3.3.13, if $\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P}$ then $\dashv_{\texttt{BiInt}} \tau(\mathcal{S}\ \Gamma \ \not\Rightarrow\ \Delta\ \mathcal{P})$. That is, there exists a `BiInt` model $M = \langle W, \leq, V\rangle$ and a world $w \in W$ such that $w \Vdash \bigvee \mathcal{S}, \Gamma$ and $w \dashv\!\mid \Delta, \bigwedge \mathcal{P}$. Then clearly $w \Vdash \Gamma$ and $w \dashv\!\mid \Delta$, that is, $\dashv_{\texttt{BiInt}} \hat{\Gamma} \to \check{\Delta}$. Q.E.D.

For full completeness, we also need decidability, which we establish next.

3.4 Decision procedure and complexity

In this and the subsequent section, we concentrate only on **GBiInt**, that is, the sequent calculus without cut and special rules, but with the derived transitional rules $\to_{R2}$ and $\prec_{L2}$. We show that **GBiInt** is terminating and gives a decision procedure for `BiInt`. Indeed, we can even create **GBiInt** trees automatically, using backward search.

We start with $\Gamma \overset{?}{\Rightarrow} \Delta$, where $\overset{?}{\Rightarrow}$ is unknown, and we want to determine whether $\overset{?}{\Rightarrow} = \Rightarrow$ or $\overset{?}{\Rightarrow} = \not\Rightarrow$. We apply the rules of **GBiInt** backwards, using the systematic procedure outlined in Figure 3.2. When the recursive calls of the procedure return, they replace $\overset{?}{\Rightarrow}$ with either $\Rightarrow$ or $\not\Rightarrow$, depending on the derivability/refutability of the subtrees and thus the appropriate rule form. At the end, our rules will deliver either $\emptyset\ \Gamma\ \Rightarrow\ \Delta\ \emptyset$ or $\mathcal{S}\ \Gamma\ \not\Rightarrow\ \Delta\ \mathcal{P}$.

We first outline a simple blocking condition to ensure termination.

Definition 3.4.1 (Blocking condition)**.** *Let ρ be a rule with $n \geq 1$ premises π_i, for $1 \leq i \leq n$, and conclusion γ. Apply ρ backwards only if: $\forall \pi_i.(LHS_{\pi_i} \not\subseteq LHS_\gamma$ or $RHS_{\pi_i} \not\subseteq RHS_\gamma)$.*

Intuitively, the general blocking condition of Definition 3.4.1 states that we apply a rule backwards only if the application results in a premise which is a strict superset of the conclusion, for otherwise we would be in a loop. This simple condition, the persistence property of `BiInt` and the contractions built into our static rules ensures termination, as we show later in this section.

Remark 3.4.2. *Note that Definition 3.4.1 implicitly includes the following:*

$$\textit{if } \rho = \to_{R2} \textit{ then } \forall \Pi \in \mathcal{P}.\Pi \not\subseteq RHS_\gamma \tag{3.4.1}$$

$$\textit{if } \rho = \prec_{L2} \textit{ then } \forall \Sigma \in \mathcal{S}.\Sigma \not\subseteq LHS_\gamma \tag{3.4.2}$$

From a backward search perspective, the only difference between $\to_{R2}$, $\prec_{L2}$ and the other rules is that we must receive the variables from the left-most premise, before we can determine whether or not to create the right premises and fully apply the rule.

The intuition behind the classification of the logical rules in Section 3.2.2 is that backwards applications of static rules add formulae to the current world in the counter-model, transitional rules $\to_{R1}$ and $\prec_{L1}$ create new worlds and add formulae to them, transitional rules $\to_{R2}$ and $\prec_{L2}$ update existing worlds with new interaction formulae received from successors/predecessors, and the transitional rule Refute moves back towards the root of the counter-model when successors/predecessors do not return any new information. The classification justifies the search strategy defined in Figure 3.2.

Definition 3.4.3. *For a `BiInt`-formula A, the subformulae $sf(A)$ are defined as usual. The subformulae of an extended `BiInt`-formula $\bigvee \mathcal{Q}$ or $\bigwedge \mathcal{Q}$ are the subformulae of its members.*

To show that the procedure in Figure 3.2 gives us a decision procedure, we need to show that for all input $\Gamma \overset{?}{\Rightarrow} \Delta$, it terminates and returns either true, meaning $\vdash \emptyset\ \Gamma\ \Rightarrow\ \Delta\ \emptyset$, or false, meaning $\vdash \mathcal{S}\ \Gamma\ \not\Rightarrow\ \Delta\ \mathcal{P}$. We show termination first. So let $m = |sf(\Gamma \cup \Delta)|$ in the following definitions and lemmas.

```
Function Decide
Input: sequent $\pi_0 = \Gamma \overset{?}{\Rightarrow} \Delta$
Output: true (meaning $\vdash \emptyset\ \Gamma\ \Rightarrow\ \Delta\ \emptyset$) or false (meaning $\vdash \mathcal{S}\ \Gamma\ \not\Rightarrow\ \Delta\ \mathcal{P}$)

  1. If $\rho \in \{id, \bot_L, \top_R\}$ is applicable to $\pi_0$ then return true
  2. Else if $\rho =$ Ret is applicable to $\pi_0$ then return false
  3. Else if $\rho$ is a static rule that is applicable to $\pi_0$ then
       (a) Let $\pi_1, \cdots, \pi_n$ be the premises of $\rho$ obtained from $\pi_0$
       (b) Return $\bigwedge_{i=1}^{n} Decide(\pi_i)$
  4. Else if $Decide(\pi_1)$ = true for some premise instance $\pi_1$ obtained from $\pi_0$ via
     $\rho = \rightarrow_{R1}$ then return true
  5. Else if $Decide(\pi_1)$ = true for some premise instance $\pi_1$ obtained from $\pi_0$ via
     $\rho = \prec_{L1}$ then return true
  6. Else if $Decide(\pi)$ = false for some left premise instance $\pi$ obtained from $\pi_0$ via
     $\rho = \rightarrow_{R2}$ and condition 3.4.1 is met then
       (a) Let $\pi_i$ for $1 \leq i \leq n$ and $n \geq 1$ be the right premises of $\rho$ obtained from $\pi_0$
       (b) Return $\bigwedge_{i=1}^{n} Decide(\pi_i)$
  7. Else if $Decide(\pi)$ = false for some left premise instance $\pi$ of obtained from $\pi_0$
     via $\rho = \prec_{L2}$ and condition 3.4.2 is met then
       (a) Let $\pi_i$ for $1 \leq i \leq n$ and $n \geq 1$ be the right premises of $\rho$ obtained from $\pi_0$
       (b) Return $\bigwedge_{i=1}^{n} Decide(\pi_i)$
  8. Else $\rho =$ Refute must be applicable to $\pi_0$. Apply $\rho$ and return false.
  9. Endif
```

We have left out the variables for simplicity, but in each return statement it is implicit that the variables are returned as specified in the conclusion of the rules defined in Section 3.2.2. Also, $\bigwedge_{i=1}^{n} Decide(\pi_i)$ is true iff $Decide(\pi_i)$ is true for all premises π_i for $1 \leq i \leq n$.

Figure 3.2: A proof search strategy for **GBiInt**

Definition 3.4.4 (LEN). *Let $>_{len}$ be a lexicographic ordering of sequents:*

$$(\Gamma_2 \overset{?}{\Rightarrow} \Delta_2) >_{len} (\Gamma_1 \overset{?}{\Rightarrow} \Delta_1) \text{ iff } [(|\Gamma_2| > |\Gamma_1|) \text{ or } (|\Gamma_2| = |\Gamma_1| \text{ and } |\Delta_2| > |\Delta_1|)].$$

Definition 3.4.5. *We use the following terms:*

- successor rules *to refer to* $\rightarrow_{R1}$, $\rightarrow_{R2}$ *and* Refute.
- predecessor rules *to refer to* $\prec_{L1}$, $\prec_{L2}$ *and* Refute.

- successor premises *to refer to the premise of* $\rightarrow_{R1}$, *the left premise of* $\rightarrow_{R2}$, *and the* $\rightarrow$*-premises of* Refute.
- predecessor premises *to refer to the premise of* $\prec_{L1}$, *the left premise of* $\prec_{L2}$, *and the* $\prec$*-premises of* Refute.
- transitional premises *to refer to both predecessor premises and successor premises.*

Definition 3.4.6. *Given a* **GBiInt***-tree* $\mathcal{T}$ *and a branch or part thereof* $\mathcal{B}$ *in* $\mathcal{T}$, *we say that* $\mathcal{B}$ *is* successor-only *if* $\mathcal{B}$ *contains only applications of static rules,* $\rightarrow_{R1}$, $\rightarrow_{R2}$, *and successor premises. Similarly,* $\mathcal{B}$ *is* predecessor-only *if* $\mathcal{B}$ *contains only applications of static rules,* $\prec_{L1}$, $\prec_{L2}$, *and predecessor premises. A branch is* single-directional *if it is successor-only or predecessor-only. Finally, a branch contains a* direction switch *if it is not single-directional.*

Lemma 3.4.7. *Every single-directional branch of every* **GBiInt***-tree is* $\mathcal{O}(m^2)$ *long.*

Proof. We prove only the successor-only case since the predecessor-only case is symmetric.

We show that on every successor-only branch, the length of a sequent defined via $>_{len}$ increases with every rule application, and that it can increase $\mathcal{O}(m^2)$ times.

Consider a rule ρ, and a backwards application of ρ to some $\Gamma \overset{?}{\Rightarrow} \Delta$, which yields n premises $\Gamma_i \overset{?}{\Rightarrow} \Delta_i$, where $1 \leq i \leq n$. If ρ is a static rule, then for all premises i, we have $(\Gamma_i \overset{?}{\Rightarrow} \Delta_i) >_{len} (\Gamma \overset{?}{\Rightarrow} \Delta)$ from the generalised blocking condition (Definition 3.4.1). We only show the case for $\rho = \rightarrow_{R1}$ since the other cases are similar:

Case $\rho = \rightarrow_{R1}$: The principal formula is $A \rightarrow B$. Consider the premise $\Gamma_1 \overset{?}{\Rightarrow} \Delta_1$. According to our strategy, the $\rightarrow^I_R$ rule has already been applied and thus $B \in \Delta$, so $\rightarrow_{R1}$ is applied only if $A \notin \Gamma$. Therefore, for the premise, we have $|\Gamma_1| > |\Gamma|$.

The length of a sequent can increase either by adding a subformula to the LHS, or by keeping the LHS unchanged and adding a subformula to the RHS. We can add a subformula to the LHS at most m times. After each such addition, the length of the RHS either remains the same (if a static rule was applied) or decreases to 1 (if $\rightarrow_{R1}$, $\rightarrow_{R2}$ or Refute was applied). In the latter case, we can again add a subformula to the RHS at most $m - 1$ times. Hence the length can increase $\mathcal{O}(m^2)$ times. Q.E.D.

Definition 3.4.8 (Degree)**.** *The degree of a* `BiInt` *formula* A *is the number of* $\rightarrow$ *and* $\prec$ *connectives in* A. *The degree of a sequent* $\Gamma \overset{?}{\Rightarrow} \Delta$ *is defined as:*

$$deg(\Gamma \overset{?}{\Rightarrow} \Delta) = \sum_{A \in sf(\Gamma \cup \Delta)} deg(A)$$

Corollary 3.4.9. *By the subformula property, the degree of a sequent cannot increase in backward search. For any sequents* γ_1 *and* γ_2, $deg(\gamma_2) < deg(\gamma_1)$ *if* $sf(\gamma_2) \subsetneq sf(\gamma_1)$.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\pi_1 = (A_1 \overset{?}{\Rightarrow} B_1, \Delta_1)}}{\Gamma_1, A_1 \mathrel{-\!\!<} B_1 \overset{?}{\Rightarrow} \Delta_1} \; {\scriptstyle \mathrel{-\!\!<}_{L1}}}{\vdots}}{\Gamma_0, A_0 \overset{?}{\Rightarrow} B_0}}{\pi_0 = (\Gamma_0 \overset{?}{\Rightarrow} \Delta_0, A_0 \rightarrow B_0)} \; {\scriptstyle \rightarrow_{R1}}
$$

$$\vdots$$

Figure 3.3: **GBiInt** switching premises

That is, removing some formula A from a sequent during backward search decreases the degree of the sequent if A is not a subformula of any other formula in the sequent since A no longer contributes to the sum of degrees of subformulae.

Lemma 3.4.10. *Every rule of* **GBiInt** *has a finite number of premises.*

Proof. Obvious for all rules except $\rightarrow_{R2}$ and $\mathrel{-\!\!<}_{L2}$. For $\rightarrow_{R2}$ and $\mathrel{-\!\!<}_{L2}$, the number of premises is $1 + n$, where n is the number of sets in the variable $\mathcal{S}$ or $\mathcal{P}$ of the left premise. But both $\mathcal{S}$ and $\mathcal{P}$ are subsets of the powerset of $sf(\Gamma \cup \Delta)$ of the end sequent $\Gamma \overset{?}{\Rightarrow} \Delta$. Therefore, each of $\mathcal{S}$ and $\mathcal{P}$ are of finite size $\mathcal{O}(2^m)$, where $m = |sf(\Gamma \cup \Delta)|$. Q.E.D.

Lemma 3.4.11. *Let $\mathcal{B}$ be any branch of any* **GBiInt** *tree that contains a direction switch, and let π_0 be the conclusion of a successor (resp. predecessor) rule and let π_1 be the premise of a predecessor (resp. successor) rule. Then $deg(\pi_1) < deg(\pi_0)$.*

Proof. We do the case where $\mathcal{B}$ contains $\rightarrow_{R1}$ and $\mathrel{-\!\!<}_{L1}$: see Figure 3.3. An inspection of the rules in Section 3.2.2 shows that expanding π_0 during backward search using $\rightarrow_{R2}$ with the principal formula $A_0 \rightarrow B_0$ yields the same successor premise $\Gamma_0, A_0 \overset{?}{\Rightarrow} B_0$ as using $\rightarrow_{R1}$. Similarly for the corresponding $\rightarrow$-premise of Refute, and symmetrically for predecessor premises. Thus all other cases of direction switches are equivalent from a backward search perspective.

Let $C \in sf(\pi_0)$ be some formula such that $deg(C) = max(\{deg(A) \mid A \in sf(\pi_0)\})$: that is, C is one of the subformulae with the maximum degree. In particular, this means that C is not a subformula of any formula with a larger degree. We shall now show that $C \notin sf(\pi_1)$.

There are two cases:

$C \notin sf(\Gamma_0)$**:** Then $C \in sf(\Delta_0)$ or $C = A_0 \rightarrow B_0$. In both cases, $C \notin sf(\pi_1)$.

$C \in sf(\Gamma_0)$**:** Then $C \in sf(A_1)$ or $C \in sf(B_1)$ implies $deg(A_1 \mathrel{-\!\!<} B_1) > deg(C)$, contradicting our assumption that $deg(C) = max(\{deg(A) \mid A \in sf(\pi_0)\})$. Therefore, either:

- C and all its occurrences in subformulae disappear from the sequent at the premise of $\prec_{L1}$, in which case $C \notin sf(\pi_1)$, or
- C is moved to the RHS of the sequent by applying the $\rightarrow_L$ rule to some formula $C \rightarrow D \in sf(\Gamma_0)$. However, since $deg(C \rightarrow D) > deg(C)$, this again contradicts our assumption that $deg(C) = max(\{deg(A) \mid A \in sf(\pi_0)\})$.

We have shown that for some formula C we have $C \in sf(\pi_0)$ and $C \notin sf(\pi_1)$. Also, by the subformula property of **GBiInt** we have $sf(\pi_1) \subseteq sf(\pi_0)$. Together with $C \in sf(\pi_0)$ and $C \notin sf(\pi_1)$, this means $sf(\pi_1) \subsetneq sf(\pi_0)$. Then by Corollary 3.4.9 we have $deg(\pi_1) < deg(\pi_0)$. Note that the steps indicated by vertical ellipses (dots) in Figure 3.3 are arbitrary, since by Corollary 3.4.9 no rule can increase the degree of a sequent. Since $deg(\pi_1) < deg(\pi_0)$, every direction switch must decrease the degree of the sequent. Q.E.D.

Lemma 3.4.12. *Any branch in any* **GBiInt** *tree built via the strategy of Figure 3.2 is* $\mathcal{O}(m^3)$ *long.*

Proof. By Lemma 3.4.7, we can move in one direction $\mathcal{O}(m^2)$ times, before we must stop or change direction. By Lemma 3.4.11, every direction change decreases the degree of the sequent. We can change direction $\mathcal{O}(m)$ times since the degree of the end sequent is $\left(\sum_{A \in sf(\Gamma \cup \Delta)} deg(A)\right) = \mathcal{O}(m)$. Thus, every branch has length $\mathcal{O}(m^2) \times \mathcal{O}(m) = \mathcal{O}(m^3)$. Q.E.D.

Theorem 3.4.13 (Termination)**.** *Every* **GBiInt***-tree built via the strategy of Figure 3.2 is finite.*

Proof. By Lemmas 3.4.10 and 3.4.12, every tree is finitely branching, and every branch is finite. Q.E.D.

Note that the strategy of Figure 3.2 is required for both completeness and termination. In particular, transitional rules must be applied only to saturated sequents as this blocks the transitional rules from creating an infinite branch by repeatedly using the same formula as the principal formula. The other aspects of the strategy are required for completeness.

Theorem 3.4.14 (Decision procedure)**.** *For every* $\Gamma \overset{?}{\Rightarrow} \Delta$*, there is an effective decision procedure to decide whether* $\vdash \emptyset\ \Gamma \ \Rightarrow \ \Delta\ \emptyset$ *or* $\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow \ \Delta\ \mathcal{P}$*, where* $\mathcal{S} \subseteq 2^{sf(\Gamma \cup \Delta)}$ *and* $\mathcal{P} \subseteq 2^{sf(\Gamma \cup \Delta)}$.

Proof. By Theorem 6.4.6, the backward search procedure of Figure 3.2 terminates for all Γ, Δ. It is clear that cases 1 to 8 of Figure 3.2 are exhaustive, thus it is fully deterministic and always returns an answer of either true ($\vdash \emptyset\ \Gamma \ \Rightarrow \ \Delta\ \emptyset$) or false ($\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow \ \Delta\ \mathcal{P}$). Q.E.D.

We can now obtain traditional completeness and its "dual" as corollaries:

Corollary 3.4.15 (Completeness)**.** *If* $\models_{\texttt{BiInt}} \hat{\Gamma} \rightarrow \check{\Delta}$ *then* $\vdash \emptyset\ \Gamma \ \Rightarrow \ \Delta\ \emptyset$.

Proof. Suppose $\models_{\texttt{BiInt}} \hat{\Gamma} \rightarrow \check{\Delta}$. Run our procedure on $\Gamma \overset{?}{\Rightarrow} \Delta$, and obtain by Theorem 3.4.14 that either $\vdash \emptyset\ \Gamma \ \Rightarrow \ \Delta\ \emptyset$ or $\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow \ \Delta\ \mathcal{P}$. In the first case we are done, since we have shown what was required. In the second, Theorem 3.3.15 gives us that $\models\!\!\!|_{\texttt{BiInt}} \hat{\Gamma} \rightarrow \check{\Delta}$. But this contradicts our assumption that $\models_{\texttt{BiInt}} \hat{\Gamma} \rightarrow \check{\Delta}$. Hence the second case is impossible. Q.E.D.

Corollary 3.4.16. *If* $\models\!\!\!|_{\texttt{BiInt}} \hat{\Gamma} \rightarrow \check{\Delta}$ *then* $\vdash \mathcal{S}\ \Gamma \ \not\Rightarrow \ \Delta\ \mathcal{P}$ *for some* $\mathcal{S}$ *and* $\mathcal{P}$.

Proof. Symmetric to the proof of Corollary 3.4.15. Q.E.D.

Lemma 3.4.17. *A* **GBiInt** *sequent takes* $\mathcal{O}(2^m)$ *space.*

Proof. A **GBiInt** sequent $\mathcal{S}\ \Gamma \ \overset{?}{\Rightarrow} \ \Delta\ \mathcal{P}$ consists of 4 components. Each of Γ and Δ are of size $\mathcal{O}(m)$, and each of the variables $\mathcal{S}$ and $\mathcal{P}$ are subsets of the powerset of $sf(\Gamma \cup \Delta)$. Therefore, each of $\mathcal{P}$ and $\mathcal{S}$ are of size $\mathcal{O}(2^m)$, and the overall sequent is of size $\mathcal{O}(2^m)$. Q.E.D.

Theorem 3.4.18 (Complexity)**.** *Our decision procedure* **GBiInt** *takes* $\mathcal{O}(2^m)$ *space.*

Proof. Since our decision procedure performs depth-first construction/traversal of **GBiInt** trees, it suffices to show that any path of a **GBiInt** tree takes $\mathcal{O}(2^m)$ space. By Lemma 3.4.12, any path is at most of polynomial length, and by Lemma 3.4.17, each sequent on such a path uses at most exponential space. Therefore, any path of a **GBiInt** tree takes $\mathcal{O}(2^m)$ space. Q.E.D.

Given a graph, a cluster is a set of nodes which form a strongly connected component. A cluster is proper if it contains more than one node. A BiInt frame is rooted if there exists a root world w such that every world u can be reached from w by following $\leq$-edges or $\leq^{-1}$-edges. The next corollary follows directly from termination and from our construction in the proof of Lemma 3.3.7 since we never create proper clusters, i.e., we do not reuse worlds.

Corollary 3.4.19. BiInt *is characterised by finite rooted reflexive and transitive frames with no proper clusters.*

An exponential-space decision procedure for a PSPACE problem may seem suboptimal, especially since a PSPACE decision procedure for a very similar logic [68] already exists. But the situation is not that simple as we show next. While the initial algorithm given by Horrocks et al. is indeed in PSPACE , it is too inefficient in practice due to the many restarts: *"the technique is not used in practice as rebuilding the discarded parts of the completion tree can be very costly"* [65]. In fact, the usual approach is to implement decision procedures for logics with inverse roles without the depth-first strategy, but this makes it necessary to *"save the state of the whole completion tree at each* $\vee$*-rule application"* [65], which is similar to our encoding of the open branches using sets-of-sets. Thus practical decision procedures do not necessarily have to be optimal, meaning that our suboptimal approach for BiInt is not as bad as it may appear.

Finally, the space usage of our algorithm is amenable to optimisation. We can make a number of observations about how the formulae in the variables are passed down from leaves towards the root, and store them in more efficient data structures than sets of sets. For example, an approach like frequent-pattern trees [61] can be used to efficiently store sets containing overlapping elements. Additionally, it is likely that some of the traditional optimisations for tableau calculi [67] are still applicable in the intuitionistic case.

3.5 Comparison with related work

3.5.1 Other sequent calculi for `BiInt`

As mentioned before, Uustalu has recently given a counter-example [95] to Rauszer's cut-elimination theorem [99]. Uustalu's counterexample also shows that Crolard's sequent calculus [27] for `BiInt` is not cut-free. Uustalu's counterexample fails in both Rauszer's and Crolard's calculi because they limit certain sequent rules to singleton succedents or antecedents in the conclusion, and the rules do not capture the "forward" and "backward" interaction between implication and exclusion.

Pinto and Uustalu have recently given a cut-free sequent-calculus for `BiInt` [95]. Their calculus uses labelled formulae, thereby utilising some semantic aspects, such as explicit worlds and accessibility, directly in the rules. On the other hand, our calculus **GBiInt** is purely syntactic and our variables $\mathcal{S}$ and $\mathcal{P}$ have no semantic content, although they clearly have some proof/refutation search content. Nevertheless, some aspects of our work bear similarities to Pinto and Uustalu's work. In particular, our termination proof relies on the fact that no branch of a **GBiInt•** derivation can have an infinite number of direction switches: essentially the same result is shown by Pinto and Uustalu for their proof search procedure [95].

If we were interested only in decision procedures, we could obtain a decision procedure for `BiInt` by embedding it into the tense logic `Kt.S4` [121], and using tableaux for description logics with inverse roles [68]. However, an embedding into `Kt.S4` provides no proof-theoretic insights into `BiInt` itself. Moreover, the restart technique of Horrocks et al. [68] involves non-deterministic expansion of disjunctions, which is complicated by inverse roles. Their actual implementation avoids this non-determinism by keeping a global view of the whole counter-model under construction. In contrast, we handle this non-determinism by syntactically encoding it using variables and extended sequents. Moreover, the persistence and reverse persistence properties of `BiInt` are not present in classical modal or tense logics. These properties of `BiInt` are in fact very helpful in developing our decision procedure, in particular the saturation process.

3.5.2 Comparison with other calculi for `Int`

Recall that `LJT` [36] is a traditional sequent calculus for `Int`, and `CRIP` [94] is a refutation calculus for `Int`. Although we developed **GBiInt** independently from these calculi, we now compare the three calculi.

In LJT and in other traditional sequent calculi, one relates the syntactic judgement of derivability to the semantic judgement of validity by showing that the rules preserve validity downwards (if the premises are valid, then the conclusion is valid). Similarly, in CRIP, one shows that the rules preserve falsifiability downwards (if the premises are falsifiable then the conclusion is falsifiable). Both are local notions referring to a single rule. For completeness or sufficiency in traditional sequent calculi, one typically shows that a counter-model can be constructed from a failed proof search, referring to a global notion of failure.

Because **GBiInt** contains both derivations and refutations, we show that **GBiInt** rules preserve the generalised semantic judgement (either validity or falsifiability) downwards. In addition, the side conditions in some of our refutation rules incrementally encode aspects of proof search failure that allow us to directly construct a counter-model from a refutation; thus proof search and refutation search are interleaved in our decision procedure. Then all we need to show for completeness or sufficiency is that for every input sequent, our calculus will produce either a derivation or a refutation.

Since the Refute rule of **GBiInt** serves a similar function to rule (11) in CRIP [94], our proof of Lemma 3.3.7 bears similarities to a part of the counter-model construction for CRIP. Indeed, if we were to restrict **GBiInt** to the Int (or the DualInt) fragment of BiInt, we would obtain a calculus whose refutation part is similar to CRIP, with the major difference being the termination mechanism. CRIP and LJT achieve termination using four separate contraction-free implication left rules each of which inspects the structure of the formula A in $A \to B$. A previous attempt to extend this idea to the case where A is of the form $C \prec D$ was unsuccessful [32]. Thus the technique of inspecting the form of A in $A \to B$ and B in $A \prec B$ is unlikely to succeed for BiInt. In a certain sense, this is because in Int/DualInt the transitional rules always create successors/predecessors. That is, there is no interaction as there is in BiInt. Additionally, because BiInt requires contraction in the implication left and exclusion right rules for completeness, our termination mechanism relies on the implicit contractions in all our static rules and the generalised blocking conditions.

3.5.3 Other termination mechanisms

There are other ways to obtain a terminating sequent calculus for Int using contraction-free calculi [36] or history methods [63; 71]. However, as explained above, contraction-free methods are less suitable when the interaction between Int and DualInt formulae needs to be considered, since they erase potentially relevant formulae too soon during backward proof search.

Our saturation-based termination mechanism, which relies on the persistence property of BiInt, may be seen as a replacement for Heuerding's history-based loop checks. But we found it easier to prove semantic completeness with our method than with history-based methods since both Heuerding et al. [63] and Howe [71] prove completeness using syntactic transformations of derivations. Dynamic blocking [68] is another possible solution for termination, but it also requires histories in a sequent calculus setting.

Part II

Towards taming proof search in display logic

Chapter 4

A shallow inference nested sequent calculus for bi-intuitionistic logic

We begin our quest for a proof search method for display logic by using bi-intuitionistic logic as our first case study. This chapter gives two shallow inference calculi for `BiInt`, while the next one gives deep inference calculi. The shallow calculi $\mathbf{LBiInt_1}$ and $\mathbf{LBiInt_2}$ which we present now sit somewhere in between display calculi and traditional sequent calculi in terms of cut-elimination and proof-search:

$\mathbf{LBiInt_1}$: The calculus $\mathbf{LBiInt_1}$ shares some features of display calculi, in that it has certain structural rules that allow shuffling of structures in a sequent, akin to the display postulates used in display calculi to display a formula in a structure. The syntactic judgments in $\mathbf{LBiInt_1}$ can be seen as a tree of (traditional) sequents, and the structural rules can be used to "display" a sequent by bringing it to the root of an equivalent tree. The logical rules of $\mathbf{LBiInt_1}$ are similar to those in Gentzen's traditional sequent calculus, as they apply only to the topmost sequent in the tree of sequents. The virtue of $\mathbf{LBiInt_1}$ is twofold: its contraction and weakening rules can be restricted to formulae while its purely syntactic cut-elimination proof is simple and very similar to the cut-elimination proof for display calculi.

$\mathbf{LBiInt_2}$: The calculus $\mathbf{LBiInt_2}$ is a refinement of $\mathbf{LBiInt_1}$ and is obtained by absorbing all the structural rules of $\mathbf{LBiInt_1}$ into the logical rules. The calculus $\mathbf{LBiInt_2}$ is easily shown to be sound, since its rules are derivable in $\mathbf{LBiInt_1}$. But from a proof-search perspective, we are able to associate a terminating and systematic backward proof-search strategy for applying the rules of $\mathbf{LBiInt_2}$. The idea behind backward proof search for $\mathbf{LBiInt_2}$ is that the introduction rules for implication and subtraction can be used to 'suspend' proof search of a (top-level) sequent and to 'restart' it at a later stage. Such restart rules are already known in the literature, but as far as we are aware, our work is the first time they have been given a purely proof-theoretic setting.

We do not have a direct syntactic proof of completeness of $\mathbf{LBiInt_2}$ with respect to $\mathbf{LBiInt_1}$. Instead, we prove the semantic completeness of $\mathbf{LBiInt_2}$ directly by

$$\begin{array}{lcllcl}
\tau^-(A) & = & A & \tau^+(A) & = & A \\
\tau^-(X,Y) & = & \tau^-(X) \wedge \tau^-(Y) & \tau^+(X,Y) & = & \tau^+(X) \vee \tau^+(Y) \\
\tau^-(X \triangleright Y) & = & \tau^-(X) {-\!\!<} \tau^+(Y) & \tau^+(X \triangleright Y) & = & \tau^-(X) \to \tau^+(Y)
\end{array}$$

$$\tau(X \Rightarrow Y) = \tau^-(X) \to \tau^+(Y)$$

Figure 4.1: Formula translation of nested sequents

showing how to construct a counter-model from a failed proof search attempt in **LBiInt₂**.

Sections 4.1 to 4.3 present the calculus $\mathbf{LBiInt_1}$ and its meta theory via theorems on cut elimination, soundness and completeness. While the structural rules in $\mathbf{LBiInt_1}$ are somewhat more restrictive than display calculi, and hence reduce slightly the non-determinism arising from the structural rules of display calculi, they still pose some difficulty in proof search. In Section 4.4, we present a restricted version of $\mathbf{LBiInt_1}$, called $\mathbf{LBiInt_2}$, in which all the structural rules are omitted and are instead absorbed into logical rules. In the same section we also give a terminating proof search strategy for $\mathbf{LBiInt_2}$, and a direct semantic completeness proof of $\mathbf{LBiInt_2}$.

Note. Some of the results of this chapter have been published in [57].

4.1 The sequent calculus $\mathbf{LBiInt_1}$

Our methodology is to use nested sequents which are similar to the structures in display calculi but which are more restricted than those used in display calculi. In particular, not all the display structural connectives used in Goré's display calculus for `BiInt` [53] are allowed and certain display postulates are missing. The idea is to get as close as possible to sequent calculus, because then we may be able to use the standard saturation techniques for proof search common in sequent calculus. Chapter A in the Appendix provides a more detailed comparison between $\mathbf{LBiInt_1}$ and Goré's calculus.

A structure is defined by the following grammar, where A is a `BiInt` formula:

$$X := \emptyset \mid A \mid (X, X) \mid X \triangleright X.$$

The structural connective "," is associative and commutative and $\emptyset$ is its unit. We always consider structures modulo these equivalences. To reduce parentheses, we assume that "," binds tighter than "$\triangleright$". Thus, we write $X, Y \triangleright Z$ to mean $(X, Y) \triangleright Z$.

If X and Y are structures, then $X \Rightarrow Y$ is a *nested shallow sequent*. The structural connective comma "," is a proxy for conjunction (on the left) and disjunction (on the right), while $\triangleright$ is a proxy for exclusion (on the left) and implication (on the right): see Figure 4.1 for a formula-translation of nested sequents. Note that our structures are a

Identity and cut:

$$\frac{}{X, A \Rightarrow A, Y}\ id \qquad \frac{X_1 \Rightarrow Y_1, A \quad A, X_2 \Rightarrow Y_2}{X_1, X_2 \Rightarrow Y_1, Y_2}\ cut$$

Structural rules:

$$\frac{X \Rightarrow Y}{X, A \Rightarrow Y}\ w_L \qquad \frac{X \Rightarrow Y}{X \Rightarrow A, Y}\ w_R \qquad \frac{X, A, A \Rightarrow Y}{X, A \Rightarrow Y}\ c_L \qquad \frac{X \Rightarrow A, A, Y}{X \Rightarrow A, Y}\ c_R$$

$$\frac{(X_1 \triangleright Y_1), X_2 \Rightarrow Y_2}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_L \qquad \frac{X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R$$

$$\frac{X_2 \Rightarrow Y_2, Y_1}{X_2 \triangleright Y_2 \Rightarrow Y_1}\ \triangleright_L \qquad \frac{X_1, X_2 \Rightarrow Y_2}{X_1 \Rightarrow X_2 \triangleright Y_2}\ \triangleright_R$$

Logical rules:

$$\frac{X, B_i \Rightarrow Y}{X, B_1 \wedge B_2 \Rightarrow Y}\ \wedge_L\ i \in \{1,2\} \qquad \frac{X \Rightarrow A, Y \quad X \Rightarrow B, Y}{X \Rightarrow A \wedge B, Y}\ \wedge_R$$

$$\frac{X, A \Rightarrow Y \quad X, B \Rightarrow Y}{X, A \vee B \Rightarrow Y}\ \vee_L \qquad \frac{X \Rightarrow B_i, Y}{X \Rightarrow B_1 \vee B_2, Y}\ \vee_R\ i \in \{1,2\}$$

$$\frac{X \Rightarrow A, Y \quad X, B \Rightarrow Y}{X, A \rightarrow B \Rightarrow Y}\ \rightarrow_L \qquad \frac{X, A \Rightarrow B}{X \Rightarrow Y, A \rightarrow B}\ \rightarrow_R$$

$$\frac{A \Rightarrow B, Y}{X, A \prec B \Rightarrow Y}\ \prec_L \qquad \frac{X \Rightarrow A, Y \quad X, B \Rightarrow Y}{X \Rightarrow A \prec B, Y}\ \prec_R$$

Figure 4.2: **LBiInt₁**: a shallow inference nested sequent calculus for `BiInt`

simplification of Goré's structures, since we overload the "$\triangleright$" connective by interpreting it differently in positive and negative contexts, just as the structural connective "," can be overloaded to represent both disjunction and conjunction in different contexts.

In order to formally describe the "zooming-in" to a substructure within a nested sequent, we will use the notion of a context. A *context* is a structure with a hole or a placeholder $[]$. Contexts are ranged over by $\Sigma[]$. We write $\Sigma[X]$ for the structure obtained by filling the hole $[]$ in the context $\Sigma[]$ with a structure X.

A *simple* context is defined via:

$$\Sigma[] ::= [] \mid \Sigma[], (Y) \mid (Y), \Sigma[]$$

Intuitively, the hole in a simple context is never under the scope of $\triangleright$. The hole in a simple context is of *neutral* polarity. Positive and negative contexts are defined inductively as follows:

- If $\Sigma[]$ is a simple context then $\Sigma[] \triangleright Y$ is a negative context and $Y \triangleright \Sigma[]$ is a positive context.

- If $\Sigma[]$ is a positive/negative context then so are $(\Sigma[], Y)$ and $(Y, \Sigma[])$ and $\Sigma[] \triangleright Y$ and $Y \triangleright \Sigma[]$.

The hole in a negative context has negative polarity, and the hole in a positive context has positive polarity. We write $\Sigma^-[]$ to indicate that $\Sigma[]$ is a negative context and $\Sigma^+[]$ to indicate that it is a positive context.

Note that our definition of polarities is non-traditional since further nesting within $\triangleright$ *does not* change polarity. As we explain later, this becomes useful when we want to nest and un-nest $X \triangleright Y$-substructures of sequents: the $\triangleright$ connective is effectively a nested turnstile.

Example 4.1.1. *The context* $[], (X \triangleright Y)$ *is a simple context but* $([], X) \triangleright Y$ *is not.* $(([], X) \triangleright Y) \triangleright Z$ *is a negative context and* $(X \triangleright Y, []) \triangleright Z$ *is a positive context.*

A k-hole context is a context with k holes. Given a k-hole context $Z[\cdots]$ we write $Z[X^k]$ to stand for the structure obtained from $Z[\cdots]$ by replacing each hole with an occurrence of the structure X. A k-hole context is positive if every hole in it has positive polarity, and it is *quasi-positive* if every hole in it is either neutral or positive. A k-hole context is negative if every hole in it has negative polarity, and it is *quasi-negative* if every hole in it is either neutral or negative.

Example 4.1.2. *The 3-hole context* $[], ([] \triangleright Y) \triangleright ([] \triangleright Z)$ *is quasi-negative. The 2-hole context* $(Y \triangleright []) \triangleright (Z \triangleright [])$ *is positive.*

Our first sequent calculus $\mathbf{LBiInt_1}$ for bi-intuitionistic logic is given in Figure 4.2. The introduction rules for the logical connectives are the standard ones. The logical rules $\to_L$ for implication on the left and $-\!\!<_R$ for exclusion on the right are non-invertible, since they lose structures or formulas going upwards. Since we have contraction and weakening, on both sides of the sequent, it is possible to formulate invertible logical rules by implicit contraction, as we shall see later. $\mathbf{LBiInt_1}$ is very similar to the display calculus for bi-intuitionistic logic of Goré [51], but with some differences:

- The contraction and the weakening rules are applicable to formulae only, not structures in general like in display calculi. But we shall see that the general contraction and weakening rules are derivable from the "atomic" ones in $\mathbf{LBiInt_1}$, which is not the case for Goré's system.

- The structural rules s_L and s_R are more general than the display postulates in display logic. These rules are derivable in Goré's system, but one needs to use contraction and weakening on structures. We give the detailed derivation in Lemma A.1.4 in the Appendix.

As a consequence of these differences, cut elimination for **LBiInt$_1$** does not necessarily follow from cut elimination for its display calculus counterpart. However, it may be possible to modify Goré's system in such a way that there is a mapping between the cut free proofs of both **LBiInt$_1$** and the modified calculus. We leave the details of such a connection to future work.

Display property. A nested sequent can be seen as a tree of traditional sequents. The structural rules of **LBiInt$_1$** allow shuffling of structures to display/un-display a particular node in the tree, so inference rules can be applied to it. This is similar to the display property in traditional display calculi, where any substructure can be displayed and un-displayed. We state the display property of **LBiInt$_1$** more precisely in subsequent lemmas. We shall use two "display" rules which are easily derivable using s_L, s_R, $\triangleright_L$ and $\triangleright_R$; double lines indicate that the rules may be applied both top-to-bottom and vice versa:

$$\frac{(X_1 \triangleright X_2) \Rightarrow Y}{X_1 \Rightarrow X_2, Y}\; rp_L^{\triangleright} \qquad \frac{X_1 \Rightarrow (X_2 \triangleright Y)}{X_1, X_2 \Rightarrow Y}\; rp_R^{\triangleright}$$

Let $DP = \{rp_R^{\triangleright}, rp_L^{\triangleright}\}$ and let DP-derivable mean "derivable using rules only from DP".

Lemma 4.1.3 (Display property for simple contexts)**.** *Let $\Sigma[]$ be a simple context. Let X be a structure and p a propositional variable not occurring in X nor $\Sigma[]$. Then there exist structures Y and Z such that:*

1. *$Y \Rightarrow p$ is DP-derivable from $X \Rightarrow \Sigma[p]$ and*
2. *$p \Rightarrow Z$ is DP-derivable from $\Sigma[p] \Rightarrow X$.*

Proof. By induction on the size of the context $\Sigma[]$. The non-trivial cases are when $\Sigma = \Sigma_1[], (W)$ or $\Sigma = (W), \Sigma_1[]$. We give the required derivations for the first case; the second case is analogous due to the commutativity of comma.

1. We first obtain the following derivation:

$$\frac{X \Rightarrow \Sigma_1[p], W}{X \triangleright W \Rightarrow \Sigma_1[p]}\; rp_L^{\triangleright}$$

   Now we apply the induction hypothesis to the smaller context $\Sigma_1[]$ to obtain the required derivation:

$$\begin{array}{c}\dfrac{X \Rightarrow \Sigma_1[p], W}{X \triangleright W \Rightarrow \Sigma_1[p]}\; rp_L^{\triangleright} \\ \vdots \\ Y \Rightarrow p\end{array}$$

2. We first obtain the following derivation:

$$\frac{\Sigma_1[p], W \Rightarrow X}{\Sigma_1[p] \Rightarrow W \triangleright X}\; rp_R^{\triangleright}$$

Now we apply the induction hypothesis to the smaller context $\Sigma_1[]$ to obtain the required derivation:

$$\dfrac{\Sigma_1[p], W \Rightarrow X}{\Sigma_1[p] \Rightarrow W \triangleright X}\, rp_R^{\triangleright}$$
$$\vdots$$
$$p \Rightarrow Z$$

Q.E.D.

Lemma 4.1.4 (Display property for positive contexts)**.** *Let $\Sigma[]$ be a positive context. Let X be a structure and p a propositional variable not occurring in X nor $\Sigma[]$. Then there exist structures Y and Z such that:*

1. *$Y \Rightarrow p$ is DP-derivable from $X \Rightarrow \Sigma[p]$, and*

2. *$Z \Rightarrow p$ is DP-derivable from $\Sigma[p] \Rightarrow X$*

Proof. We prove both statements simultaneously by induction on the size of the context $\Sigma[]$. The non-trivial cases are when $\Sigma[] = W \triangleright \Sigma_1[]$ or $\Sigma[] = \Sigma_1[] \triangleright W$. The following are the required derivations:

1. • Case when $\Sigma[] = W \triangleright \Sigma_1[]$. We first obtain the following derivation:

$$\dfrac{X \Rightarrow (W \triangleright \Sigma_1[p])}{X, W \Rightarrow \Sigma_1[p]}\, rp_R^{\triangleright}$$

If $\Sigma_1[]$ is simple context, we apply Lemma 4.1.3 to $X, W \triangleright \Sigma_1[p]$, otherwise $\Sigma_1[]$ is a positive context and we apply statement (1) of the induction hypothesis to $X, W \triangleright \Sigma_1[p]$. In both cases, we obtain the required derivation:

$$\dfrac{X \Rightarrow (W \triangleright \Sigma_1[p])}{X, W \Rightarrow \Sigma_1[p]}\, rp_R^{\triangleright}$$
$$\vdots$$
$$Y \Rightarrow p$$

• Case when $\Sigma[] = \Sigma_1[] \triangleright W$. We first obtain the following derivation:

$$\dfrac{\dfrac{X \Rightarrow (\Sigma_1[p] \triangleright W)}{X, \Sigma_1[p] \Rightarrow W}\, rp_R^{\triangleright}}{\Sigma_1[p] \Rightarrow (X \triangleright W)}\, rp_R^{\triangleright}$$

Here $\Sigma_1[]$ must be a positive context, so we apply statement (2) of the induction hypothesis to $\Sigma_1[] \Rightarrow (X \rhd W)$ and obtain the required derivation:

$$\begin{array}{c} \dfrac{\dfrac{X \Rightarrow (\Sigma_1[p] \rhd W)}{X, \Sigma_1[p] \Rightarrow W}\, rp^{\rhd}_{R}}{\Sigma_1[p] \Rightarrow (X \rhd W)}\, rp^{\rhd}_{R} \\ \vdots \\ Z \Rightarrow p \end{array}$$

2. • Case when $\Sigma[] = W \rhd \Sigma_1[]$. We first obtain the following derivation:

$$\dfrac{\dfrac{(W \rhd \Sigma_1[p]) \Rightarrow X}{W \Rightarrow \Sigma_1[p], X}\, rp^{\rhd}_{L}}{(W \rhd X) \Rightarrow \Sigma_1[p]}\, rp^{\rhd}_{L}$$

If $\Sigma_1[]$ is simple context, we apply Lemma 4.1.3 to $(W \rhd X) \Rightarrow \Sigma_1[p]$, otherwise $\Sigma_1[]$ is a positive context and we apply statement (1) of the induction hypothesis to $(W \rhd X) \Rightarrow \Sigma_1[p]$. In both cases, we obtain the required derivation:

$$\begin{array}{c} \dfrac{\dfrac{(W \rhd \Sigma_1[p]) \Rightarrow X}{W \Rightarrow \Sigma_1[p], X}\, rp^{\rhd}_{L}}{(W \rhd X) \Rightarrow \Sigma_1[p]}\, rp^{\rhd}_{L} \\ \vdots \\ Y \Rightarrow p \end{array}$$

• Case when $\Sigma[] = \Sigma_1[] \rhd W$. We first obtain the following derivation:

$$\dfrac{(\Sigma_1[p] \rhd W) \Rightarrow X}{\Sigma_1[p] \Rightarrow W, X}\, rp^{\rhd}_{L}$$

Here $\Sigma_1[]$ must be a positive context, so we apply statement (2) of the induction hypothesis to $\Sigma_1[p] \Rightarrow W, X$ and obtain the required derivation:

$$\begin{array}{c} \dfrac{(\Sigma_1[p] \rhd W) \Rightarrow X}{\Sigma_1[p] \Rightarrow W, X}\, rp^{\rhd}_{L} \\ \vdots \\ Z \Rightarrow p \end{array}$$

Q.E.D.

Lemma 4.1.5 (Display property for negative contexts)**.** *Let $\Sigma[]$ be a negative context. Let X be a structure and p a propositional variable not occurring in X nor $\Sigma[]$. Then there exist structures Y and Z such that:*

1. *$p \Rightarrow Y$ is DP-derivable from $X \Rightarrow \Sigma[p]$ and*
2. *$p \Rightarrow Z$ is DP-derivable from $\Sigma[p] \Rightarrow X$.*

Proof. Symmetric to the proof of Lemma 4.1.4. Q.E.D.

Note that since the rules in DP are all invertible, the derivations constructed in the above lemmas are invertible derivations. That is, we can derive $Y \Rightarrow p$ from $X \Rightarrow \Sigma[p]$ and vice versa. Note also that since the rules in DP are closed under substitution, this also means $Y \Rightarrow Z$ is derivable from $X \Rightarrow \Sigma[Z]$, and vice versa, for any Z.

The display property of pure display calculi is the ability to display/un-display a particular structure with respect to a top-level turnstile $\vdash$ (say) as the *whole* of the antecedent or succedent. For example, we have to display $V \triangleright W$ as the whole of the antecedent or succedent as $V \triangleright W \vdash Z$ or $Z \vdash V \triangleright W$.

Our nested sequent calculus instead enables us to "zoom in" to $V \triangleright W$ in $X \Rightarrow Y$ by explicitly transforming the latter into $X', V \Rightarrow W, Y'$ so we can apply a rule to any top-level formula/structure of V or W. Indeed, this is why the notion of polarity in our nested sequent calculus is non-traditional: when we display a structure $V \triangleright W$ at the top level as $X', V \Rightarrow W, Y'$, we want V to be negative and W to be positive regardless of how deep $V \triangleright W$ is nested.

The following two propositions state the admissibility of the general contraction and weakening rules. These can be proved by using the structural rules s_L, s_R, $\triangleright_L$ and $\triangleright_R$.

Proposition 4.1.6 (Admissibility of general contraction)**.** *The two contraction rules shown below are cut-free admissible in* $\mathbf{LBiInt_1}$*:*

$$\frac{X, Y, Y \Rightarrow Z}{X, Y \Rightarrow Z}\ gc_L \qquad \frac{X \Rightarrow Y, Y, Z}{X \Rightarrow Y, Z}\ gc_R$$

Proof. We prove this simultaneously by induction on the size of Y. We show a derivation of the gc_L rule; the case for gc_R is symmetric. The non-trivial case is when $Y = Y_1 \triangleright Y_2$. We show that in this case, the contraction rule can be reduced to contractions on smaller structures, which therefore are admissible by the induction hypothesis (IH):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{X, (Y_1 \triangleright Y_2), (Y_1 \triangleright Y_2) \Rightarrow Z}{(Y_1 \triangleright Y_2), (Y_1 \triangleright Y_2) \Rightarrow X \triangleright Z}\ \triangleright_R}{(Y_1 \triangleright Y_2), Y_1 \Rightarrow Y_2, (X \triangleright Z)}\ s_L}{Y_1, Y_1 \Rightarrow Y_2, Y_2, (X \triangleright Z)}\ s_L}{Y_1, Y_1 \Rightarrow Y_2, (X \triangleright Z)}\ gc_R\ (IH)}{Y_1 \Rightarrow Y_2, (X \triangleright Z)}\ gc_L\ (IH)}{Y_1 \triangleright Y_2 \Rightarrow X \triangleright Z}\ \triangleright_L}{X, (Y_1 \triangleright Y_2) \Rightarrow Z}\ s_R$$

Q.E.D.

Proposition 4.1.7 (Admissibility of general weakening)**.** *The two weakening rules below* *are cut-free admissible in* **LBiInt$_1$***:*

$$\frac{X \Rightarrow Z}{X, Y \Rightarrow Z}\, gw_L \qquad \frac{X \Rightarrow Z}{X \Rightarrow Y, Z}\, gw_R$$

Proof. We prove this simultaneously by induction on the size of Y. We show a derivation of the gw_L rule; the case for gw_R is symmetric. The non-trivial case is when $Y = Y_1 \rhd Y_2$. We show that in this case, the weakening rule can be reduced to weakening on smaller structures, which therefore are admissible by the induction hypothesis (IH):

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{X \Rightarrow Z}{\Rightarrow (X \rhd Z)}\, \rhd_R}{\Rightarrow Y_2, (X \rhd Z)}\, gw_R\ (IH)}{Y_1 \Rightarrow Y_2, (X \rhd Z)}\, gw_L\ (IH)}{(Y_1 \rhd Y_2) \Rightarrow (X \rhd Z)}\, \rhd_L}{X, (Y_1 \rhd Y_2) \Rightarrow Z}\, s_R$$

Q.E.D.

Proposition 4.1.8. *The id rule can be restricted to the atomic form:*

$$\frac{}{X, p \Rightarrow p, Y}\, id$$

Proof. We show that the general *id* rule is derivable given the atomic *id* rule, that is, for any `BiInt` formula A, there is an **LBiInt$_1$** derivation of $X, A \Rightarrow A, Y$ whose leaves are instances of the atomic *id* rule. We prove this by induction on the length of A; the base case when A is an atom is trivial. We show two example cases; the others are similar or easier.

- Case when $A = C \wedge D$. We construct the following derivation, where the induction hypothesis gives us that $X, C \Rightarrow C, Y$ and $X, D \Rightarrow D, Y$ have derivations whose leaves are instances of the atomic *id* rule.

$$\cfrac{\cfrac{X, C \Rightarrow C, Y}{X, C \wedge D \Rightarrow C, Y}\, \wedge_L \qquad \cfrac{X, D \Rightarrow D, Y}{X, C \wedge D \Rightarrow D, Y}\, \wedge_L}{X, C \wedge D \Rightarrow C \wedge D, Y}\, \wedge_R$$

- Case when $A = C \to D$.

$$\cfrac{\cfrac{\cfrac{X, C \Rightarrow C, Y}{X, C \Rightarrow C, D, Y}\, w_R \qquad \cfrac{X, D \Rightarrow D, Y}{X, C, D \Rightarrow D, Y}\, w_L}{X, C \to D, C \Rightarrow D, Y}\, \to_L}{X, C \to D \Rightarrow C \to D, Y}\, \to_R$$

Q.E.D.

So from now on, we assume that all *id* rules are of the atomic form.

$$\underset{\Pi}{\dfrac{\overset{\Pi_1}{X_1 \Rightarrow Y_1, p} \quad \overset{\Pi_2}{p, X_2 \Rightarrow Y_2}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ cut}$$

$$\underset{\Pi_1}{\begin{array}{c} \dfrac{}{p \Rightarrow p}\ id \quad \cdots \quad \dfrac{}{p \Rightarrow p}\ id \\ \vdots \\ X_1 \Rightarrow Y_1, p \end{array}} \qquad \underset{\Pi'}{\begin{array}{c} \dfrac{\overset{\Pi_2}{p, X_2 \Rightarrow Y_2}}{p \Rightarrow (X_2 \triangleright Y_2)}\ \triangleright_R \quad \cdots \quad \dfrac{\overset{\Pi_2}{p, X_2 \Rightarrow Y_2}}{p \Rightarrow (X_2 \triangleright Y_2)}\ \triangleright_R \\ \vdots \\ \dfrac{X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R \end{array}}$$

Figure 4.3: Cut-elimination example

4.2 Cut elimination

Since our calculi are not display calculi, Belnap's general cut elimination theorem [12] cannot be used directly to prove cut elimination for our calculi. One way of showing cut elimination for $\mathbf{LBiInt_1}$ would be an indirect proof via a detour through display calculus. That is, one first designs a corresponding display calculus for $\mathbf{LBiInt_1}$ for which Belnap's cut elimination theorem can be used, e.g., by modifying Goré's calculus to work with a more restricted form of structures, and then showing that the cut free proofs of this display calculus can be mapped to cut-free proofs of $\mathbf{LBiInt_1}$. We show here a simple and direct cut elimination proof instead.[1]

Although the proof system $\mathbf{LBiInt_1}$ shares some similarity with traditional Gentzen systems, cut elimination for $\mathbf{LBiInt_1}$ as presented here follows a different technique from the standard cut elimination technique for sequent calculus. In particular, when the cut formula is not principal in either one of the premises of the cut rule, no cut reductions are required in our cut elimination proof. Instead, the structural rules s_L and s_R allow us to carry the context of one premise of the cut to its other premise resulting in a "proof substitution" akin to the normalisation proofs in natural deduction. Apart from Belnap's cut-elimination proof for display logic, the closest technique we know of is the cut elimination proof for classical logic in a proof system using deep inference [16].

For example, suppose we have a derivation which ends with an instance of cut on the atom p, as illustrated in the top of Figure 4.3. Suppose also that Π_1 is the cut-free derivation on the bottom left in Figure 4.3 where the occurrence of p in the root sequent participates in n instances of *id* in the leaves of Π_1.

Then a cut free derivation Π' for $X_1, X_2 \Rightarrow Y_1, Y_2$ can be obtained by replacing the instances of the cut formula p in Π_1 with the structure $(X_2 \triangleright Y_2)$ and replacing the

[1]The cut-elimination proof in this chapter is due to Alwen Tiu, and is included in this thesis for completeness.

leaves of Π_1, where the cut formula p is used, with the derivation Π_2. This cut-free derivation is schematically presented on the right in Figure 4.3. Note that the parts of derivations indicated by vertical and horizontal ellipses remain the same, except for the substitution of p by $X_2 \triangleright Y_2$.

The reductions for the cases where the cut formula is non-atomic follow essentially the same idea. That is, we substitute the cut formula in one premise of the cut rule with the context of the other premise, and expand this context when the cut formula is used. The only difference is that in the case of non-atomic cut formula, we need to produce extra cuts to make this substitution work. But all the cuts produced are of smaller size, therefore the whole process terminates.

In the following, we write $|A|$ for the *size* of the formula A: the number of logical operators appearing in A. In an instance of a cut rule

$$\frac{X_1 \Rightarrow Y_1, A \quad A, X_2 \Rightarrow Y_2}{X_1, X_2 \Rightarrow Y_1, Y_2}\ cut$$

the formula A is called the *cut formula* of the cut instance. The *cut-rank* of the cut instance is $|A|$. Given a derivation Π, we denote with $mc(\Pi)$ the maximum of the cut-ranks in Π. If there are no cuts in Π then $mc(\Pi) = 0$.

Lemma 4.2.1 states the proof substitutions needed to eliminate atomic cuts.

Lemma 4.2.1. *Suppose $p, X \Rightarrow Y$ is cut-free derivable for some fixed p, X and Y. Then for any k-hole positive context $Z_1[\cdots]$ and any l-hole quasi-positive context $Z_2[\cdots]$, if $Z_1[p^k] \Rightarrow Z_2[p^l]$ is cut-free derivable, then $Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ is cut-free derivable.*

Proof. Let Π be a cut-free derivation of $p, X \Rightarrow Y$ and let Θ be a cut-free derivation of $Z_1[p^k] \Rightarrow Z_2[p^l]$. We construct a cut-free derivation Θ' of $Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ by induction on the height of Θ. Most cases follow straightforwardly from the induction hypothesis. The only non-trivial case is when p is active in the derivation, i.e., when Θ ends with an *id* rule or a contraction rule applied to an occurrence of p to be substituted for:

- Suppose Θ is
$$\frac{}{Z_1'[p^k], p \Rightarrow p, Z_2'[p^{l-1}]}\ id$$
Note that the p immediately to the left of the turnstile cannot be part of the p^k by the restrictions on the context $Z_1[\cdots]$. The derivation Θ' is then constructed as follows, where we use dashed lines to abbreviate derivations:
$$\dfrac{\dfrac{\overset{\Pi}{p, X \Rightarrow Y}}{p \Rightarrow (X \triangleright Y)}\ \triangleright_R}{Z_1'[(X \triangleright Y)^k], p \Rightarrow (X \triangleright Y), Z_2'[(X \triangleright Y)^{l-1}]}\ gw_R; gw_L$$

- Suppose Θ is

$$\dfrac{\begin{array}{c}\Theta_1\\ Z_1[p^k] \Rightarrow p, p, Z_2'[p^{l-1}]\end{array}}{Z_1[p^k] \Rightarrow p, Z_2'[p^{l-1}]}\ c_R$$

By induction hypothesis, we have a cut-free derivation Θ_1' of

$$Z_1[(X \triangleright Y)^k] \Rightarrow (X \triangleright Y), (X \triangleright Y), Z_2'[(X \triangleright Y)^{l-1}].$$

The derivation Θ' is then constructed as follows:

$$\dfrac{\begin{array}{c}\Theta_1'\\ Z_1[(X \triangleright Y)^k] \Rightarrow (X \triangleright Y), (X \triangleright Y), Z_2'[(X \triangleright Y)^{l-1}]\end{array}}{Z_1[(X \triangleright Y)^k] \Rightarrow (X \triangleright Y), Z_2'[(X \triangleright Y)^{l-1}]}\ gc_R$$

Note that gw_R and gc_R and gw_L are cut-free derivable in $\mathbf{LBiInt_1}$ by Proposition 4.1.6 and Proposition 4.1.7. Q.E.D.

Lemmas 4.2.2-4.2.6 state the proof substitutions needed for non-atomic cuts.

Lemma 4.2.2. *Let Θ be a derivation of*

$$Z_1[(A_1 \vee A_2)^k] \Rightarrow Z_2[(A_1 \vee A_2)^l]$$

for some k-hole quasi-negative context $Z_1[\cdots]$ and l-hole negative context $Z_2[\cdots]$, such that $mc(\Theta) < |A_1 \vee A_2|$. Let Π_i be a derivation of $X \Rightarrow Y, A_i$, for some $i \in \{1,2\}$, such that $mc(\Pi_i) < |A_1 \vee A_2|$. Then there is a derivation Θ' with $mc(\Theta') < |A_1 \vee A_2|$ of

$$Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l].$$

Proof. By induction on the height of Θ. In the following, we let $A = A_1 \vee A_2$. Most cases follow straightforwardly from the induction hypothesis. The only interesting case is when a left-rule is applied to an occurrence of $A_1 \vee A_2$ which is to be replaced by $X \triangleright Y$. That is, Θ is

$$\dfrac{\begin{array}{c}\Theta_1\\ Z_1'[A^{k-1}], A_1 \Rightarrow Z_2[A^l]\end{array} \quad \begin{array}{c}\Theta_2\\ Z_1'[A^{k-1}], A_2 \Rightarrow Z_2[A^l]\end{array}}{Z_1'[A^{k-1}], A_1 \vee A_2 \Rightarrow Z_2[A^l]}\ \vee_L$$

By induction hypothesis, we have a derivation Θ_i', for each $i \in \{1,2\}$, of

$$Z_1'[(X \triangleright Y)^{k-1}], A_i \Rightarrow Z_2[(X \triangleright Y)^l]$$

with $mc(\Theta'_i) < |A_1 \vee A_2|$. The derivation Θ' is then constructed as follows:

$$\dfrac{\dfrac{\overset{\Pi_i}{X \Rightarrow Y, A_i}}{X \triangleright Y \Rightarrow A_i}\triangleright_L \quad \overset{\Theta'_i}{Z'_1[(X \triangleright Y)^{k-1}], A_i \Rightarrow Z_2[(X \triangleright Y)^l]}}{Z'_1[(X \triangleright Y)^{k-1}], (X \triangleright Y) \Rightarrow Z_2[(X \triangleright Y)^l]}\ cut$$

Q.E.D.

Lemma 4.2.3. *Let Θ be a derivation of*

$$Z_1[(A_1 \wedge A_2)^k] \Rightarrow Z_2[(A_1 \wedge A_2)^l]$$

for some k-hole quasi-negative context $Z_1[\cdots]$ and l-hole negative context $Z_2[\cdots]$ with $mc(\Theta) < |A_1 \wedge A_2|$. Let Π_1 be a derivation of $X \Rightarrow Y, A_1$ and let Π_2 be a derivation of $X \Rightarrow Y, A_2$ with $mc(\Pi_1) < |A_1 \wedge A_2|$ and $mc(\Pi_2) < |A_1 \wedge A_2|$. Then there is a derivation Θ' with $mc(\Theta') < |A_1 \wedge A_2|$ of

$$Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l].$$

Proof. Analogous to the proof of Lemma 4.2.2. Q.E.D.

Lemma 4.2.4. *Let Θ be a derivation of*

$$Z_1[(A \to B)^k] \Rightarrow Z_2[(A \to B)^l]$$

for some k-hole quasi-negative context $Z_1[\cdots]$ and l-hole negative context $Z_2[\cdots]$ with $mc(\Theta) < |A \to B|$. Let Π be a derivation of $X, A \Rightarrow B$ with $mc(\Pi) < |A \to B|$. Then there is a derivation Θ' with $mc(\Theta') < |A \to B|$ of

$$Z_1[X^k] \Rightarrow Z_2[X^l].$$

Proof. By induction on the height of Θ. As in the previous lemmas, the non-trivial case is when Θ ends with $\to_L$ on $A \to B$:

$$\dfrac{\overset{\Theta_1}{Z'_1[(A \to B)^{k-1}] \Rightarrow A, Z_2[(A \to B)^l]} \quad \overset{\Theta_2}{Z'_1[(A \to B)^{k-1}], B \Rightarrow Z_2[(A \to B)^l]}}{Z'_1[(A \to B)^{k-1}], A \to B \Rightarrow Z_2[(A \to B)^l]}\ \to_L$$

By induction hypothesis, we have derivations Θ'_1 and Θ'_2 respectively of the sequents below where $mc(\Theta'_1) < |A \to B|$ and $mc(\Theta'_2) < |A \to B|$:

$$Z'_1[X^{k-1}] \Rightarrow A, Z_2[X^l] \qquad\qquad Z'_1[X^{k-1}], B \Rightarrow Z_2[X^l]$$

In the following, we let V_1 denote $Z'_1[X^{k-1}]$ and V_2 denote $Z_2[X^l]$. The derivation Θ'

is constructed as follows:

$$\dfrac{\dfrac{\begin{matrix}\Theta_1' \\ V_1 \Rightarrow A, V_2\end{matrix} \qquad \dfrac{\begin{matrix}\Pi \\ X, A \Rightarrow B\end{matrix} \qquad \begin{matrix}\Theta_2' \\ V_1, B \Rightarrow V_2\end{matrix}}{V_1, A, X \Rightarrow V_2}\, cut}{V_1, V_1, X \Rightarrow V_2, V_2}\, cut}{V_1, X \Rightarrow V_2}\, gc_L; gc_R$$

Q.E.D.

Lemma 4.2.5. *Let Θ be a derivation of*

$$Z_1[(A \prec B)^k] \Rightarrow Z_2[(A \prec B)^l]$$

for some k-hole quasi-negative context $Z_1[\cdots]$ and l-hole negative context $Z_2[\cdots]$ with $mc(\Theta) < |A \prec B|$. Let Π_1 be a derivation of $X \Rightarrow Y, A$ and let Π_2 be a derivation of $X, B \Rightarrow Y$ with $mc(\Pi_1) < |A \prec B|$ and $mc(\Pi_2) < |A \prec B|$. Then there is a derivation Θ' with $mc(\Theta') < |A \prec B|$ of

$$Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l].$$

Proof. The non-trivial case is when Θ ends with $\prec_L$ on $A \prec B$:

$$\dfrac{\begin{matrix}\Theta_1 \\ A \Rightarrow B, Z_2[(A \prec B)^l]\end{matrix}}{Z_1'[(A \prec B)^{k-1}], A \prec B \Rightarrow Z_2[(A \prec B)^l]}\, \prec_L$$

By induction hypothesis, we have a derivation Θ_1' of

$$A \Rightarrow B, Z_2[(X \triangleright Y)^l]$$

with $mc(\Theta_1') < |A \prec B|$. Let V denote the structure $Z_2[(X \triangleright Y)^l]$. Then Θ' is constructed as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{matrix}\Pi_1 \\ X \Rightarrow Y, A\end{matrix} \qquad \dfrac{\begin{matrix}\Theta_1' \\ A \Rightarrow B, V\end{matrix} \qquad \begin{matrix}\Pi_2 \\ X, B \Rightarrow Y\end{matrix}}{A, X \Rightarrow Y, V}\, cut}{X, X \Rightarrow Y, Y, V}\, cut}{X \Rightarrow Y, V}\, gc_L; gc_R}{X \triangleright Y \Rightarrow V}\, \triangleright_L}{Z_1'[(X \triangleright Y)^{k-1}], X \triangleright Y \Rightarrow V}\, gw_L$$

Q.E.D.

Lemma 4.2.6. *Let Θ be a derivation of $Z_1[A^k] \Rightarrow Z_2[A^l]$ where A is a non-atomic formula, $Z_1[\cdots]$ is a k-hole positive context, $Z_2[\cdots]$ is an l-hole quasi-positive context, and $mc(\Theta) < |A|$. Let Π be a derivation of $A, X \Rightarrow Y$ with $mc(\Pi) < |A|$. Then there is a derivation Θ' with $mc(\Theta') < |A|$ of $Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$.*

Proof. By induction on the height of Θ and case analysis on A. The non-trivial case is when Θ ends with a right-introduction rule on A. That is, in this case, we have $Z_2[A^l] = (Z_2'[A^{l-1}], A)$ for some quasi-positive context $Z_2'[\cdots]$. We distinguish several cases depending on A. We show here the cases where A is either a disjunction $C \vee D$, or an implication $C \to D$.

- Suppose $A = C \vee D$ and Θ is the following derivation:

$$\dfrac{\begin{array}{c}\Theta_1\\ Z_1[(C \vee D)^k] \Rightarrow Z_2'[(C \vee D)^{l-1}], C\end{array}}{Z_1[(C \vee D)^k] \Rightarrow Z_2'[(C \vee D)^{l-1}], C \vee D}\ \vee_R$$

  By induction hypothesis, we have a derivation Θ_1' of

$$Z_1[(X \triangleright Y)^k] \Rightarrow Z_2'[(X \triangleright Y)^{l-1}], C$$

  such that $mc(\Theta_1') < |C \vee D|$. Let $W_1 = Z_1[(X \triangleright Y)^k]$ and let $W_2 = Z_2[(X \triangleright Y)^{l-1}]$. Applying Lemma 4.2.2 to Π and Θ_1', we obtain a derivation θ of

$$(W_1 \triangleright W_2), X \Rightarrow Y$$

  such that $mc(\theta) < |C \vee D|$. The derivation Θ' is then constructed as follows:

$$\dfrac{\dfrac{\begin{array}{c}\Theta_1'\\ (W_1 \triangleright W_2), X \Rightarrow Y\end{array}}{W_1 \triangleright W_2 \Rightarrow X \triangleright Y}\ \triangleright_R}{W_1 \Rightarrow W_2, (X \triangleright Y)}\ s_L$$

  Clearly, $mc(\Theta') < |C \vee D|$.

- Suppose $A = C \to D$ and Θ is

$$\dfrac{\begin{array}{c}\Theta_1\\ Z_1[(C \to D)^k], C \Rightarrow D\end{array}}{Z_1[(C \to D)^k] \Rightarrow Z_2'[(C \to D)^{l-1}], C \to D}\ \to_R$$

  By induction hypothesis, we have a derivation Θ_1' of

$$Z_1[(X \triangleright Y)^k], C \Rightarrow D$$

  Then the derivation Θ' is constructed as follows:

$$\dfrac{\dfrac{\begin{array}{c}\theta\\ Z_1[(X \triangleright Y)^k], X \Rightarrow Y\end{array}}{Z_1[(X \triangleright Y)^k] \Rightarrow (X \triangleright Y)}\ \triangleright_R}{Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^{l-1}], (X \triangleright Y)}\ gw_R$$

  where θ is obtained by applying Lemma 4.2.4 to Π and Θ_1'.

The other cases are treated analogously, using Lemmas 4.2.3 and Lemma 4.2.5. Q.E.D.

Finally, cut elimination is proved by simple proof substitutions, the construction of which is given by the preceding lemmas.

Theorem 4.2.7. *If $X \Rightarrow Y$ is* **LBiInt$_1$***-derivable then it is also cut-free derivable.*

Proof. As typical in cut elimination proofs, we remove topmost cuts in succession. Let Π be a derivation of **LBiInt$_1$** with a topmost cut instance

$$\frac{\overset{\Pi_1}{X_1 \Rightarrow Y_1, A} \quad \overset{\Pi_2}{X_2, A \Rightarrow Y_2}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ cut$$

Note that Π_1 and Π_2 are both cut-free since this is a topmost instance in Π. We use induction on the size of A to eliminate this topmost instance of cut.

If A is an atomic formula p then the cut free derivation is constructed as follows where Θ is obtained from applying Lemma 4.2.1 to Π_2 and Π_1:

$$\frac{\overset{\Theta}{X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R$$

If A is non-atomic, using Lemma 4.2.6 we get the following derivation θ:

$$\frac{\overset{\Theta}{X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R$$

We have $mc(\theta) < |A|$ by Lemma 4.2.6, therefore by induction hypothesis, we can remove all the cuts in θ to get a cut-free derivation of $X_1, X_2 \Rightarrow Y_1, Y_2$. Q.E.D.

4.3 Soundness and completeness of LBiInt$_1$

To prove soundness, we refer to the interpretation of sequents as formulae given in Figure 4.1.

Theorem 4.3.1 (Soundness)**.** *Every* **LBiInt$_1$***-derivable formula is* `BiInt`*-valid.*

Proof. We show that for every rule ρ of **LBiInt$_1$**

$$\frac{X_1 \Rightarrow Y_1 \quad \cdots \quad X_n \Rightarrow Y_n}{X \Rightarrow Y}\ \rho$$

the following holds: if for every $i \in \{1, \ldots, n\}$, the formula $\tau^-(X_i) \rightarrow \tau^+(Y_i)$ is valid then the formula $\tau^-(X) \rightarrow \tau^+(Y)$ is valid. Since the formula-translation $(\tau^-(X) \wedge A) \rightarrow (A \vee \tau^+(Y))$ of the *id* rule is obviously valid, it then follows that every formula derivable in **LBiInt$_1$** is also valid.

For all the rules of **LBiInt₁**, except $\triangleright_L$ and $-\!\!<_L$, we can show the stronger statement that the following formula is valid:

$$[(\tau^-(X_1) \to \tau^+(Y_1)) \land \cdots \land (\tau^-(X_n) \to \tau^+(Y_n))] \to (\tau^-(X) \to \tau^+(Y)).$$

Soundness of $\triangleright_L$ and $-\!\!<_L$ are shown in the standard way, by reasoning about the forcing relation $\Vdash$ and the reflexive and transitive relation $\leq$ (recall the `BiInt` semantics we introduced in Section 2.2.4.4). For each rule, we show that if the premise of the rule is valid, then the conclusion is also valid.

We show the case for $\triangleright_L$; the case for $-\!\!<_L$ is very similar. We assume that the formula translation of the premise $X_2 \Rightarrow Y_2, Y_1$ is valid, and show that the formula translation of the conclusion $X_2 \triangleright Y_2 \Rightarrow Y_1$ is valid. That is, we assume that $\tau^-(X_2) \to (\tau^+(Y_2) \lor \tau^+(Y_1))$ is valid. This means that for every world w in every `BiInt` model $\langle M, \leq V\rangle$, we have $w \Vdash \tau^-(X_2) \to (\tau^+(Y_2) \lor \tau^+(Y_1))$. From the semantics of $\to$ (recall Figure 2.7), this means:

$$\forall\langle M, \leq V\rangle\ \forall w \in W\ \forall u \geq w.\ \text{if } u \Vdash \tau^-(X_2) \text{ then } u \Vdash (\tau^+(Y_2) \lor \tau^+(Y_1)) \tag{4.3.1}$$

Now we want to show that $(\tau^-(X_2) -\!\!< \tau^+(Y_2)) \to \tau^+(Y_1)$ is valid. We will do so by contradiction. That is, we suppose this formula is falsifiable, i.e., there exists a `BiInt` model $\langle W, \leq, V\rangle$ and a world $z' \in W$ such that $z' \Vdash \tau^-(X_2) -\!\!< \tau^+(Y_2)$ but $z' \dashv\!\vert\, \tau^+(Y_1)$. From the semantics of $-\!\!<$, this means there exists a world $u' \in W$ such that $u' \leq z'$ and $u' \Vdash \tau^-(X_2)$ and $u' \dashv\!\vert\, \tau^+(Y_2)$. By the reverse persistence property of `BiInt`, we also have that $u' \dashv\!\vert\, \tau^+(Y_1)$. Moreover, since the relation $\leq$ is reflexive, we have that $u' \geq u'$. That is, we have a world $u' \geq u'$ such that $u' \Vdash \tau^-(X_2)$ and $u' \dashv\!\vert\, \tau^+(Y_2)$ and $u' \dashv\!\vert\, \tau^+(Y_1)$. But this contradicts (4.3.1), therefore $(\tau^-(X_2) -\!\!< \tau^+(Y_2)) \to \tau^+(Y_1)$ is not falsifiable, therefore $(\tau^-(X_2) -\!\!< \tau^+(Y_2)) \to \tau^+(Y_1)$ is valid. Q.E.D.

Completeness is shown by embedding Rauszer's sequent calculus G1 [99] for `BiInt` into **LBiInt₁**. The calculus G1 contains the cut rule, and is shown to be complete by Rauszer [99]. The encoding of G1 into **LBiInt₁** is obvious since all the rules of G1 are easily derivable from the rules of **LBiInt₁**.

Theorem 4.3.2 (Completeness)**.** *Every* `BiInt`*-valid formula is* **LBiInt₁***-derivable.*

4.4 Proof search

The calculus **LBiInt₁** is not suitable for proof search, since the structural rules s_L, s_R, $\triangleright_L$ and $\triangleright_R$ can easily lead to non-termination if applied naively. In addition, we also have the usual problems with the contraction rules since they can be applied *ad infinitum*: recall the example in Figure 2.10.

We now present a refined version of **LBiInt₁**, called **LBiInt₂**, in which all the structural rules, except for $\triangleright_L$ and $\triangleright_R$, are absorbed into logical rules. The resulting calculus, for the intuitionistic fragment, resembles contraction-free calculi for the traditional Gentzen systems for intuitionistic logic, e.g., the system **G3i** in [111]. The underlying

$$\{|X|\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}$$

$$\frac{}{X, A \Rightarrow A, Y}\ id$$

$$\frac{X_2 \Rightarrow Y_2, \{|Y_1|\}}{X_1, (X_2 \triangleright Y_2) \Rightarrow Y_1}\ \triangleright_L\ \{|Y_1|\} \not\subseteq \{|Y_2|\} \qquad \frac{\{|X_1|\}, X_2 \Rightarrow Y_2}{X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)}\ \triangleright_R\ \{|X_1|\} \not\subseteq \{|X_2|\}$$

$$\frac{X, B_1 \wedge B_2, B_i \Rightarrow Y}{X, B_1 \wedge B_2 \Rightarrow Y}\ \wedge_L\ i \in \{1,2\} \qquad \frac{X \Rightarrow A \wedge B, A, Y \quad X \Rightarrow A \wedge B, B, Y}{X \Rightarrow A \wedge B, Y}\ \wedge_R$$

$$\frac{X, A \vee B, A \Rightarrow Y \quad X, A \vee B, B \Rightarrow Y}{X, A \vee B \Rightarrow Y}\ \vee_L \qquad \frac{X \Rightarrow B_1 \vee B_2, B_i, Y}{X \Rightarrow B_1 \vee B_2, Y}\ \vee_R\ i \in \{1,2\}$$

$$\frac{X, A \to B \Rightarrow A, Y \quad X, A \to B, B \Rightarrow Y}{X, A \to B \Rightarrow Y}\ \to_L \qquad \frac{X \Rightarrow Y, A \to B, B}{X \Rightarrow Y, A \to B}\ \to_{R1}$$

$$\frac{X, A \prec B, A \Rightarrow Y}{X, A \prec B \Rightarrow Y}\ \prec_{L1} \qquad \frac{X \Rightarrow A, A \prec B, Y \quad X, B \Rightarrow A \prec B, Y}{X \Rightarrow A \prec B, Y}\ \prec_R$$

$$\frac{A \Rightarrow B, \{|Y|\}, (X, A \prec B \triangleright Y)}{X, A \prec B \Rightarrow Y}\ \prec_{L2} \qquad \frac{(X \triangleright Y, A \to B), \{|X|\}, A \Rightarrow B}{X \Rightarrow Y, A \to B}\ \to_{R2}$$

Figure 4.4: **LBiInt$_2$**: a nested sequent calculus for proof search in `BiInt`

idea behind **LBiInt$_2$** is that the right-introduction rule for $\to$ and the left introduction rule for $\prec$ act as an instruction to store the current state (of proof search), and the rules $\triangleright_L$ and $\triangleright_R$ act as an instruction to restart previously stored computation states. Recall that our definition of polarities means that each structure $X \triangleright Y$, which is stored in a nested sequent, is effectively a traditional sequent in its own right, with X the negative left hand side and Y the positive right hand side.

The inference rules for **LBiInt$_2$** are given in Figure 4.4 using the notation $\{|X|\}$ to denote the *set of formulae* that appear at the top-level of X:

$$\{|X|\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}.$$

Intuitively, the set $\{|X|\}$ denotes X with all the substructures of the form $Y \triangleright Z$ or $Y \triangleright Z$ removed. For example, if X is $(A, B, (C \triangleright D))$, then $\{|X|\}$ is the set $\{A, B\}$.

The right introduction rule for $\to$ splits into two rules: $\to_{R1}$ and $\to_{R2}$. The $\to_{R1}$ rule is strictly speaking not necessary as it can be derived using $\to_{R2}$ and $\triangleright_L$. However, it is useful in our proof search strategy which relies on a *saturation* process on sequents, as we shall see later. Indeed, this rule is very similar to $\rightarrow^I_R$ from **GBiInt** in Chapter 3, and performs essentially the same function. The rule $\rightarrow_{R2}$ incorporates some features of the structural rule s_L. The left introduction rule for $-\!\!<$ splits also into two rules with roles symmetric to those for $\rightarrow$.

4.4.1 Soundness of LBiInt$_2$

For soundness of **LBiInt$_2$** we show that every **LBiInt$_2$** rule is derivable in **LBiInt$_1$**.

Theorem 4.4.1 (Soundness of **LBiInt$_2$**). *If the sequent $X \Rightarrow Y$ is derivable in* **LBiInt$_2$** *then it is also derivable in* **LBiInt$_1$**.

Proof. We show that every **LBiInt$_2$** rule is derivable in **LBiInt$_1$**. The non-trivial cases are rules $-\!\!<_{L1}$, $\rightarrow_{R1}$, $-\!\!<_{L2}$ and $\rightarrow_{R2}$. We show derivations of the rules $-\!\!<_{L2}$ (below left) and $-\!\!<_{L1}$ (below right); the other two cases are symmetric. Note that the derivation of $-\!\!<_{L1}$ uses the derived rule $-\!\!<_{L2}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow B, \{|Y|\}, (X, A-\!\!<B \rhd Y)}{A \Rightarrow B, Y, (X, A-\!\!<B \rhd Y)}\,gw_R}{A-\!\!<B \Rightarrow Y, (X, A-\!\!<B \rhd Y)}\,-\!\!<_L}{X, A-\!\!<B, A-\!\!<B \Rightarrow Y, Y}\,s_R}{X, A-\!\!<B \Rightarrow Y}\,gc_L, gc_R \qquad \dfrac{\dfrac{\dfrac{X, A-\!\!<B, A \Rightarrow Y}{A \Rightarrow (X, A-\!\!<B \rhd Y)}\,\rhd_R}{A \Rightarrow B, \{|Y|\}, (X, A-\!\!<B \rhd Y)}\,gw_R}{X, A-\!\!<B \Rightarrow Y}\,-\!\!<_{L2}$$

Q.E.D.

4.4.2 A terminating proof search strategy

We classify the rules of **LBiInt$_2$** into three groups:

Static Rules: $= \{id, \wedge_L, \wedge_R, \vee_L, \vee_R, \rightarrow_L, -\!\!<_R, -\!\!<_{L1}, \rightarrow_{R1}\}$;

Jump Rules: $= \{-\!\!<_{L2}, \rightarrow_{R2}\}$; and

Return Rules: $= \{\rhd_L, \rhd_R\}$.

We refer to the $-\!\!<_{L2}$ rule as a backward jump, and the $\rightarrow_{R2}$ rule as a forward jump.

We call a sequence of static rule applications a *saturation*.

Definition 4.4.2. *A sequent $X \Rightarrow Y$ is* saturated *iff it satisfies 1-8, and is* strongly saturated *iff it additionally satisfies 9:*

1. $\{|X|\} \cap \{|Y|\} = \emptyset$
2. *If $A \wedge B \in \{|X|\}$ then $A \in \{|X|\}$ and $B \in \{|X|\}$*
3. *If $A \wedge B \in \{|Y|\}$ then $A \in \{|Y|\}$ or $B \in \{|Y|\}$*
4. *If $A \vee B \in \{|X|\}$ then $A \in \{|X|\}$ or $B \in \{|X|\}$*

```
Function Prove
Input: sequent $\gamma_0$
Output: true (i.e. $\gamma_0$ is derivable) or false (i.e. $\gamma_0$ is not derivable)
  1. If id is applicable to $\gamma_0$ then return true
  2. Else if a static rule $\rho$ is applicable to $\gamma_0$ then
      (a) Let $\gamma_1, \cdots, \gamma_n$ be the premises of $\rho$ obtained from $\gamma_0$
      (b) Return $\bigwedge_{i=1}^{n} Prove(\gamma_i)$
  3. Else if $Prove(\gamma_1) = true$ for some premise instance $\gamma_1$ obtained from $\gamma_0$ by ap-
     plying $\rho \in \{-\!\!<_{L2}, \rightarrow_{R2}, \rhd_L, \rhd_R\}$ backward then return true
  4. Else return false.
```

Figure 4.5: A proof search strategy for $\mathbf{LBiInt_2}$

5. *If $A \vee B \in \{|Y|\}$ then $A \in \{|Y|\}$ and $B \in \{|Y|\}$*
6. *If $A \rightarrow B \in \{|X|\}$ then $A \in \{|Y|\}$ or $B \in \{|X|\}$*
7. *If $A -\!\!< B \in \{|Y|\}$ then $A \in \{|Y|\}$ or $B \in \{|X|\}$*
8. *If $A \rightarrow B \in \{|Y|\}$ then $B \in \{|Y|\}$* *If $A -\!\!< B \in \{|X|\}$ then $A \in \{|X|\}$*
9. *If $A \rightarrow B \in \{|Y|\}$ then $A \in \{|X|\}$* *If $A -\!\!< B \in \{|X|\}$ then $B \in \{|Y|\}$.*

The following definition expresses the same idea as Definition 3.4.1, but here we apply the concept to the top level formulae of the nested sequent.

Definition 4.4.3 (Generalised blocking condition)**.** *We say that an* $\mathbf{LBiInt_2}$ *rule ρ is applicable to a sequent $\gamma_0 = (X_0 \Rightarrow Y_0)$ if for every premise $X_i \Rightarrow Y_i$ of ρ we have $\{|X_i|\} \not\subseteq \{|X_0|\}$ or $\{|Y_i|\} \not\subseteq \{|Y_0|\}$.*

Thus only jump and return rules are applicable to saturated sequents. We shall show that the search strategy given in Figure 4.5 terminates, if given an input sequent with a certain simple structure, which is defined in the following.

Definition 4.4.4. *A structure is a* flat structure *if it contains no occurrences of the structural connective $\rhd$. We use Γ and Δ to stand for flat structures since flat structures can be viewed as sets of formulae. The set of (right/left)* linear structures *is the smallest set of structures that satisfies the following:*

1. *The empty structure is both a right linear structure and a left linear structure.*
2. *If X is a right (left) linear structure and Δ is a flat structure, then (X, Δ) is a right (resp. left) linear structure.*

3. *If Γ is a flat structure and X is a right linear structure, then $\Gamma \triangleright X$ is a left linear structure.*

4. *If X is a left linear structure and Δ is a flat structure, then $X \triangleright \Delta$ is a right linear structure.*

A sequent $X \Rightarrow Y$ is a linear sequent *if either X is a flat structure and Y is a right linear structure, or X is a left linear structure and Y is a flat structure.*

The intuition of Definition 4.4.4 is that a linear sequent $X \Rightarrow Y$ can take the form $(X' \triangleright Y'), \Gamma \Rightarrow \Delta$ or $\Gamma \Rightarrow \Delta, (X'' \triangleright Y'')$ or $\Gamma \Rightarrow \Delta$ where $X' \triangleright Y'$ and $X'' \triangleright Y''$ store the sequent corresponding to the previous state of computation, and Γ and Δ are sets of formulae.

Lemma 4.4.5. *Let $X \Rightarrow Y$ be a linear sequent. Then for every* **LBiInt$_2$***-derivation Π of $X \Rightarrow Y$, every sequent in Π is a linear sequent.*

Proof. Given a derivation Π of a linear sequent $X \Rightarrow Y$, we show by induction on the length of Π that every sequent in Π is a linear sequent. This is straightforward by showing that in every rule of **LBiInt$_2$**, if the conclusion of the rule is a linear sequent, then every premise of the rule is also a linear sequent, which can be verified by inspection of the rules of **LBiInt$_2$**. We give the case for the $\triangleright_L$ rule as an example:

$$\frac{X_2 \Rightarrow Y_2, \{\!|Y_1|\!\}}{X_1, (X_2 \triangleright Y_2) \Rightarrow Y_1} \triangleright_L \quad \{\!|Y_1|\!\} \not\subseteq \{\!|Y_2|\!\}$$

We assume that the conclusion $X_1, (X_2 \triangleright Y_2) \Rightarrow Y_1$ is a linear sequent and show that the premise $X_2 \Rightarrow Y_2, \{\!|Y_1|\!\}$ is a linear sequent. Since $X_1, (X_2 \triangleright Y_2)$ is not a flat structure, we have that Y_1 is a flat structure and $X_1, (X_2 \triangleright Y_2)$ is a left linear structure from Definition 4.4.4. Then we obtain the following:

1. $(X_2 \triangleright Y_2)$ is a left linear structure from Definition 4.4.4, case 2.

2. X_2 is a flat structure and Y_2 is a right linear structure from Definition 4.4.4, case 3.

3. $\{\!|Y_1|\!\}$ is a flat structure by the definition of top-level formulae of a structure.

4. $Y_2, \{\!|Y_1|\!\}$ is a right linear structure from Definition 4.4.4, case 2.

Finally, the premise $X_2 \Rightarrow Y_2, \{\!|Y_1|\!\}$ is a linear sequent from Definition 4.4.4, case 3, since X_2 is a flat structure and $Y_2, \{\!|Y_1|\!\}$ is a right linear structure. Q.E.D.

Note that as a consequence of Lemma 4.4.5, every sequent that arises during proof search for a linear sequent $X \Rightarrow Y$, using the search procedure given in Figure 4.5, is a linear sequent.

We now define a translation from linear sequents to linked lists, consisting of nodes that are pairs of sets of formulae, linked by labels marked either $\leq$ or $\geq$.

Definition 4.4.6.

$$\begin{array}{lcl} list(\Gamma \Rightarrow \Delta) & = & \langle \Gamma, \Delta \rangle \\ list((X' \rhd Y'), \Gamma \Rightarrow \Delta) & = & list(X' \Rightarrow Y') \ \leq \ \langle \Gamma, \Delta \rangle \\ list(\Gamma \Rightarrow \Delta, (X'' \rhd Y'')) & = & list(X'' \Rightarrow Y'') \ \geq \ \langle \Gamma, \Delta \rangle \end{array}$$

We write $length(L)$ to mean the number of nodes in the list L.

Corollary 4.4.7. *A backward* $\mathbf{LBiInt_2}$ *rule application to a linear sequent $X \Rightarrow Y$ can be viewed as an operation on $list(X \Rightarrow Y)$, where the conclusion (resp. premise) is the list before (resp. after) the operation. The jump rules append a node to the list, and the static rules saturate the end node. The return rules remove a node from the end of the list, and add subformulae to the penultimate node.*

For example, below left is is an instance of $\rightarrow_{R2}$ with the corresponding list structures of the premise and conclusion on the right:

$$\frac{(C \rhd B, A \rightarrow B), C, A \Rightarrow B}{C \Rightarrow B, A \rightarrow B} \rightarrow_{R2} \qquad \frac{\langle \{C\}, \{B, A \rightarrow B\} \rangle \ \leq \ \langle \{C, A\}, \{B\} \rangle}{\langle \{C\}, \{B, A \rightarrow B\} \rangle}$$

We now define a function that we will use in the termination proof.

Definition 4.4.8. *The degree of a formula is:*

$$\begin{array}{rcl} deg(p) & = & 0 \\ deg(A \wedge B) = deg(A \vee B) & = & max(deg(A), deg(B)) \\ deg(A \rightarrow B) = deg(A \prec B) & = & 1 + max(deg(A), deg(B)). \end{array}$$

The degree of a sequent is:

$$\begin{array}{rcl} deg_L(X \Rightarrow Y) & = & max\{deg(A) \mid A \in \{\!|X|\!\}\} \\ deg_R(X \Rightarrow Y) & = & max\{deg(B) \mid B \in \{\!|Y|\!\}\} \\ deg(X \Rightarrow Y) & = & max(deg_L(X \Rightarrow Y), deg_R(X \Rightarrow Y)). \end{array}$$

Note that only logical connectives contribute to these functions.

We denote with $sf(A)$ the set of subformulae of A, and

$$sf(\Gamma) = \bigcup_{A \in \Gamma} sf(A)$$

the set of subformulae of Γ. In the following, we assume that the initial input to the search procedure Prove is a linear sequent $\Gamma_0 \Rightarrow \Delta_0$, and we define $m = |sf(\Gamma_0 \cup \Delta_0)|$.

Lemma 4.4.9. *Let $X \Rightarrow Y$ be any sequent encountered during proof search. Using jump rules, $list(X \Rightarrow Y)$ can be extended at most $\mathcal{O}(m^2)$ times.*

Proof. We show that the number of jump rule applications is bounded by $\mathcal{O}(m^2)$.

First, we show that there can be at most m consecutive jumps in the same direction. In the forward case, consider an application of $\rightarrow_{R2}$ with principal formula $A \rightarrow B$.

After this application, A will be added to the LHS of the sequent, and remain on the LHS during saturation and forward jumps. Should $A \to B$ reappear on the RHS, B will be added to the RHS by the $\to_{R1}$ rule during saturation, so a repeated application of $\to_{R2}$ to $A \to B$ will be blocked by the generalised blocking condition of Definition 4.4.3. Thus since the number of $\to$-formulae is bounded by m and we can only jump on each $\to$-formula once, there can be at most m consecutive forward jumps. The backward case is symmetric.

We now show that we can switch direction at most m times. Consider a direction switch, e.g., a forward jump using $\to_{R2}$ followed by a backward jump $-\!\!<_{L2}$ (the other case is symmetric), and any static rule applications in between. Note that static rules do not increase the degree of a sequent. Let γ_0 and γ_1 be the conclusion and premise of the $\to_{R2}$ rule respectively, and let γ_2 and γ_3 be the conclusion and premise of the $-\!\!<_{R2}$ rule respectively, as shown below:

$$\cfrac{\cfrac{\vdots}{\gamma_3 = C \Rightarrow D, \Delta, ((X \triangleright Y, A \to B), \Gamma, C -\!\!< D \triangleright \Delta)}}{\gamma_2 \ = \ (X \triangleright Y, A \to B), \Gamma, C -\!\!< D \Rightarrow \Delta}\ -\!\!<_{L2}$$

$$\vdots$$

$$\cfrac{\gamma_1 \ = \ (X \triangleright Y, A \to B), \{\!|X|\!\}, A \Rightarrow B}{\gamma_0 \ = \ X \Rightarrow Y, A \to B}\ \to_{R2}$$

$$\vdots$$

Let $d_0 = deg(\gamma_0)$. We will show that $deg(\gamma_3) \leq d_0 - 1$. By inspection of the rules and Definition 4.4.8, we have the following:

$$\begin{aligned}
&deg_L(\gamma_1) \leq d_0 \\
&deg_R(\gamma_1) \leq d_0 - 1 \\
&deg_L(\gamma_2) = deg_L(\gamma_1) \leq d_0 \\
&deg_R(\gamma_2) \leq max(deg_L(\gamma_1) - 1, deg_R(\gamma_1)) = d_0 - 1 \\
&deg_L(\gamma_3) \leq deg_L(\gamma_2) - 1 = d_0 - 1 \\
&deg_R(\gamma_3) \leq max(deg_L(\gamma_2) - 1, deg_R(\gamma_2)) = d_0 - 1
\end{aligned}$$

Therefore $deg(\gamma_3) = max(deg_L(\gamma_3), deg_R(\gamma_3)) \leq d_0 - 1$.

After a direction switch, we can again make at most m jumps in one direction. Therefore the total number of jump rule applications is bounded by $\mathcal{O}(m^2)$. Q.E.D.

Note that the non-trivial part of the proof for Lemma 4.4.9 is showing that proof search cannot create infinite zig-zags of forward and backward looking edges. Thus, it is showing essentially the same result as Lemma 3.4.11 in Chapter 3, as well as that of Pinto and Uustalu's work [95]. The similarity between our work and that of Pinto and Uustalu is not surprising, given their recent work [96] on relating their labelled sequent calculus [95] to **LBiInt$_1$**. In [96], Pinto and Uustalu showed that a derivation in **LBiInt$_1$** can be translated into a derivation in their labelled sequent calculus for `BiInt` and vice versa.

Lemma 4.4.10. *Let $X \Rightarrow Y$ be any sequent encountered during proof search. Then the saturation process for $X \Rightarrow Y$ terminates after $\mathcal{O}(m)$ steps.*

Proof. Every application of a static rule adds a subformula of $sf(\Gamma_0 \cup \Delta_0)$ to the sequent. After at most m applications of static rules, the sequent will contain all subformulae of the original sequent, and hence will be saturated. Q.E.D.

Theorem 4.4.11. *The proof search strategy of Figure 4.5 terminates.*

Proof. Suppose for a contradiction that the strategy does not terminate. From Lemmas 4.4.9 and 4.4.10, we can conclude that the only way to get non-termination is for the jump and return rules to repeatedly create and remove nodes.

The length of the list is at least 1 because the first node cannot be removed. We call a node that cannot be removed *stable*. Every time a return rule removes node i from the list, it adds one or more new subformulae of $\Gamma_0 \cup \Delta_0$ to node $i-1$. After at most m such updates, node $i-1$ will contain every subformula, and the return rules will no longer be applicable to node i because their side conditions will not hold. Then node $i-1$ will become stable. Eventually all nodes will become stable, and the return rules will no longer be applicable to the end of the list. Contradiction. Q.E.D.

4.4.3 Completeness of $\mathbf{LBiInt_2}$

We will now prove that the proof search strategy of Figure 4.5 is complete with respect to `BiInt` semantics. We will do so by showing how we can use a trace of the *Prove* procedure to construct a counter-model if *Prove* returns *false*. Due to the back-propagation of formulae in `BiInt` (as well as tense logic as we shall see later), we cannot construct the counter-model by simply stitching together smaller counter-models as in basic modal logics. Therefore, rather than constructing a counter-model directly, we will use the notion of a pre-model that will contain an intermediate counter-model being constructed. Note that the pre-model construction in our proof is very similar to the restart technique for the description logic $\mathcal{ALCI}$ by Horrocks et al. [69]. In fact, $\mathbf{LBiInt_2}$ can be seen as a proof-theoretic formalisation of the restart technique used by Horrocks et al.

In the following, we use the assignment operator ":=" to emphasize the algorithmic aspect of the construction.

Definition 4.4.12. *Let Γ and Δ be sets of formulae. We say that a pair $\langle \Gamma, \Delta \rangle$ has a counter-model iff there exists a BiInt model $M = \langle W, \leq, V \rangle$ such that $w \Vdash \Gamma$ and $w \dashv\!| \Delta$ for some $w \in W$.*

We say that a node $\langle \Gamma, \Delta \rangle$ is saturated if the $\mathbf{LBiInt_2}$ sequent $\Gamma \Rightarrow \Delta$ is saturated.

Lemma 4.4.13. *For every sequent $X \Rightarrow Y$, if no $\mathbf{LBiInt_2}$ rule is applicable to $X \Rightarrow Y$, then $\langle \{\!|X|\!\}, \{\!|Y|\!\} \rangle$ has a counter-model.*

Proof. Let $W := \{w\}$, let $\leq := \{(w, w)\}$ and let $V(p) := \{w\}$ for all atoms $p \in \{\!|X|\!\}$, $V(p) := \emptyset$ otherwise. Then using the strong saturation conditions of Definition 4.4.2, we can easily show that $w \Vdash \{|X|\}$ and $w \dashv\!\vert \{Y\}$ by simultaneous induction on the length of formulae in $\{|X|\}$ and $\{|Y|\}$. Then $M = \langle W, \leq, V \rangle$ is a counter-model for $\langle \{|X|\}, \{|Y|\} \rangle$. Q.E.D.

Definition 4.4.14. *A pre-model is a tree of nodes that are pairs of sets of formulae, linked by edges labeled with $\leq$ or $\geq$, such that:*

1. *Every node is marked either C (complete) or I (incomplete).*
2. *Every internal node is marked C.*
3. *Every C-node is saturated.*
4. *A leaf node may be marked either C or I.*
5. *Every C-leaf has a counter-model.*
6. *For every two C-nodes $\langle \Gamma, \Delta \rangle$ and $\langle \Gamma', \Delta' \rangle$ such that $\langle \Gamma, \Delta \rangle \leq \langle \Gamma', \Delta' \rangle$ or $\langle \Gamma', \Delta' \rangle \geq \langle \Gamma, \Delta \rangle$, it is the case that $\Gamma \subseteq \Gamma'$ and $\Delta' \subseteq \Delta$. We say that $\langle \Gamma, \Delta \rangle$ and $\langle \Gamma', \Delta' \rangle$ are compatible in this case.*
7. *For every C-node $\langle \Gamma, \Delta \rangle$ and for every $A \!-\!\!\!< B \in \Gamma$, either $A \in \Gamma$ and $B \in \Delta$, or there exists a node $\langle \Gamma_1, \Delta_1 \rangle$ such that $\langle \Gamma, \Delta \rangle \geq \langle \Gamma_1, \Delta_1 \rangle$, and $\{A\} \subseteq \Gamma_1$ and $\Delta \cup \{B\} \subseteq \Delta_1$.*
8. *For every C-node $\langle \Gamma, \Delta \rangle$ and for every $C \to D \in \Delta$, either $C \in \Gamma$ and $D \in \Delta$, or there exists a node $\langle \Gamma_2, \Delta_2 \rangle$ such that $\langle \Gamma, \Delta \rangle \leq \langle \Gamma_2, \Delta_2 \rangle$, and $\Gamma \cup \{C\} \subseteq \Gamma_2$ and $\{D\} \subseteq \Delta_2$.*

Lemma 4.4.15. *For every pre-model M, if all leaves are marked C, then there exists a counter-model for the root of M.*

Proof. By induction on the height of M. For the base case, we use property 5 of Definition 4.4.14.

For the induction hypothesis (IH1), assume that the lemma holds for all pre-models of height $\leq k$, and consider a pre-model M of height $k+1$. Consider the root $\gamma = \langle \Gamma, \Delta \rangle$ of M. By properties 2 and 3 of Definition 4.4.14, we know that $\gamma = \langle \Gamma, \Delta \rangle$ is saturated. We obtain a counter-model $M = \langle W, \leq, V \rangle$ for γ as follows:

Let $W := \{w\}$, let $\leq := \{(w, w)\}$, and let $V(p) := \{w\}$ for every atom $p \in \Gamma$. We want to show that $w \Vdash \Gamma$ and $w \dashv\!\vert \Delta$. We do this by simultaneous induction on the length of formulae in Γ and Δ. For atoms, the valuation gives the required. For the induction hypothesis (IH2), we assume that for every formula E of length $\leq l$, if $E \in \Gamma$ then $w \Vdash E$, and if $E \in \Delta$ then $w \dashv\!\vert E$.

Consider a formula F of length $l+1$. If F is a $\wedge$- or $\vee$-formula in Γ or Δ, or an $\to$-formula in Γ or an $-\!\!\!<$-formula in Δ, we use the saturation conditions of Definition 4.4.2 and the induction hypothesis (IH2). The remaining cases are:

1. If $F = A \!-\!\!\!< B$ and $F \in \Gamma$:
   (a) By property 7 of Definition 4.4.14, either $A \in \Gamma$ and $B \in \Delta$, or there exists a node $\gamma' = \langle \Gamma_1, \Delta_1 \rangle$, such that $\langle \Gamma, \Delta \rangle \geq \langle \Gamma_1, \Delta_1 \rangle$, and $\{A\} \subseteq \Gamma_1$ and $\Delta \cup \{B\} \subseteq \Delta_1$.

(b) In the first case, the induction hypothesis (IH2) gives us $w \Vdash A$ and $w \dashv\!\mid B$. Combined with $w \leq w$, we have $w \Vdash A \prec B$.

(c) In the second case:

i. $\langle \Gamma_1, \Delta_1 \rangle$ is rooted at a pre-model M′ of height $\leq k$ such that all its leaves are marked C. By the induction hypothesis (IH1), there exists a counter-model $M' = \langle W_1, \leq_1, V_1 \rangle$ for $\langle \Gamma_1, \Delta_1 \rangle$. Let $u \in W_1$ be the world such that $u \Vdash \Gamma_1$ and $u \dashv\!\mid \Delta_1$. Then $u \Vdash A$ and $u \dashv\!\mid B, \Delta$.

ii. Let $W := W \cup W_1$, let $\leq := \leq \cup \leq_1 \cup \{(u, w)\}$, and let $V := V \cup V_1$.

iii. Since $u \Vdash A$ and $u \dashv\!\mid B, \Delta$ and $u \leq w$, we have $w \Vdash A \prec B$.

2. If $F = C \to D$ and $F \in \Delta$, then we perform a symmetric construction to Steps 1a to 1c above, using property 8 of Definition 4.4.14 and the induction hypotheses (IH1) and (IH2).

We update $\leq$ to contain the transitive closure of $\leq$. We know that M obeys persistence, because by property 6 of Definition 4.4.14, γ is compatible with each γ' used in the above construction, and each sub-model M' above obeys persistence. Therefore M is a counter-model for $\langle \Gamma, \Delta \rangle$. Q.E.D.

We now define some transformations on a pre-model that correspond to the various kinds of $\mathbf{LBiInt_2}$ rules. Given a pre-model M and nodes γ and γ_1 in M, we say that γ_1 is a *predecessor* of γ if either $\gamma_1 \leq \gamma$ or $\gamma_1 \geq \gamma$ in M.

Lemma 4.4.16. *Let M be a pre-model, and let $\gamma = \langle \Gamma, \Delta \rangle$ be some leaf I-node in M and $\gamma_1 = \langle \Gamma_1, \Delta_1 \rangle$ its predecessor. Let $X \Rightarrow Y$ be any* $\mathbf{LBiInt_2}$ *sequent such that γ is the last element of list$(X \Rightarrow Y)$, and γ_1 is the penultimate element of list$(X \Rightarrow Y)$. Let M′ be a tree obtained from M by one of the following transformations:*

(i) Let $\gamma := \langle \Gamma', \Delta' \rangle$, where $X \Rightarrow Y$ is the conclusion of a static rule application, and $\langle \Gamma', \Delta' \rangle$ contain the top-level formulae of the premise.

(ii) If γ is strongly saturated and γ_1 and γ are compatible, mark γ with C.

(iii) If γ is saturated and γ_1 and γ are compatible, mark γ with C, add an $\geq$-successor $\langle A, \{B\} \cup \Delta \rangle$ for every $A \prec B \in \Gamma$ such that $A \notin \Gamma$ or $B \notin \Delta$, and an $\leq$-successor $\langle \Gamma \cup \{C\}, D \rangle$ for every $C \to D \in \Delta$ such that $C \notin \Gamma$ or $D \notin \Delta$. Mark all the successors I.

(iv) Remove γ, mark γ_1 with I and let $\gamma_1 := \langle \Gamma_1', \Delta_1' \rangle$, where $X \Rightarrow Y$ is the conclusion of a return rule application, and $\langle \Gamma_1', \Delta_1' \rangle$ contain the top-level formulae of the premise.

Then M′ is also a pre-model that satisfies properties 1 to 8 of Definition 4.4.14.

The non-trivial cases are to show that transformations (ii), (iii) and (iv) preserve properties 3, 5 and 6.

- Property 3: we only mark γ with C using transformation (ii) if γ is strongly saturated, or using transformation (iii) if γ is saturated. Transformation (iv) resets a node to I, since the addition of formulae could potentially cause the node to become non-saturated.

- Property 5: transformation (ii) only marks a leaf node C if it is strongly saturated and compatible with its predecessor, which means that no $\mathbf{LBiInt_2}$ rule is applicable to it, and Lemma 4.4.13 gives us a counter-model. Transformation (iv) resets a node to I, since the addition of formulae could potentially cause the node to become non-saturated.

- Property 6: transformation (ii) only marks a leaf node C if it is compatible with its predecessor. Transformation (iv) resets a node to I, since the addition of formulae could potentially cause the node to become incompatible with its predecessor.

The proof of the main completeness lemma will use an induction on the height of a failed trace of *Prove*, which we define now.

Definition 4.4.17. *Let $X \Rightarrow Y$ be some sequent. A* failed trace of *Prove* for $X \Rightarrow Y$ is a tree T of sequents, linked by edges labeled with $\mathbf{LBiInt_2}$ rules, such that:

- The root of T is $X \Rightarrow Y$;

- If a call of $Prove(\gamma)$ returns at Step 4 without invoking *Prove* again, then γ is a leaf node of T;

- If a call of $Prove(\gamma)$ invokes $Prove(\gamma_i)$ at Step 3 for $1 \leq i \leq n$ and corresponding jump or static rules ρ_i, for some $n \geq 1$, and the call to each $Prove(\gamma_i)$ returns *false*, then γ is an inner node of T, and its children are γ_i, linked by labels ρ_i;

- If a call of $Prove(\gamma)$ invokes $Prove(\gamma_i)$ at Step 2 for $1 \leq i \leq 2$ and corresponding static rule ρ, and some call to $Prove(\gamma_i)$ returns *false*, then γ is an inner node of T, and its child is γ_i, linked by label ρ.

We now give the main completeness lemma. It shows that every recursive call of *Prove* that returns *false* can be used to update a pre-model, whilst retaining the required properties, and that the calls of *Prove* that return at Step 4 can be used to complete the pre-model and turn it into a proper counter-model.

Lemma 4.4.18. *Let M be some pre-model rooted at some node $\langle \Gamma_0, \Delta_0 \rangle$. If for every I-leaf $\langle \Gamma, \Delta \rangle$ of M, there exists a failed trace T of Prove for the sequent $X \Rightarrow Y$ such that $list(X \Rightarrow Y)$ is a branch $\langle \Gamma_0, \Delta_0 \rangle \cdots \langle \Gamma, \Delta \rangle$ of M, then there exists another pre-model M′ rooted at $\langle \Gamma_0', \Delta_0' \rangle$ such that all leaves of M′ are marked C and $\Gamma_0' \supseteq \Gamma_0$ and $\Delta_0' \supseteq \Delta_0$.*

Proof. Let L be the set of I-leaves in M. We prove the lemma by induction on the maximum height of traces T across all members of L.

In the base case, the height of all traces is 1. That is, for each $\langle \Gamma, \Delta \rangle$, no rules were applicable to $X \Rightarrow Y$ and *Prove* returned at Step 4. Since no rules are applicable to

$X \Rightarrow Y$, we know in particular that each $\langle \Gamma, \Delta \rangle$ is strongly saturated, and that it is compatible with its predecessor. Then we obtain M′ from M by marking each $\langle \Gamma, \Delta \rangle$ with C. By Lemma 4.4.16, using transformation (ii), M′ is a pre-model.

For the induction hypothesis, assume the lemma holds for all failed traces of *Prove* of height $\leq k$. Consider a leaf $\langle \Gamma, \Delta \rangle$ in L such that the trace T rooted at $\gamma = X \Rightarrow Y$ has height $k + 1$. For each ρ_i linking γ and γ_i for $1 \leq i \leq n$, we can use one of the transformations of Lemma 4.4.16 to obtain an M′ with leaves γ_i, such that properties 2 to 8 of Definition 4.4.14 hold in M′. In particular:

1. If $n = 1$ and ρ is a static rule, then we use transformation (i).

2. If some ρ_i is a return rule, then we use transformation (iv).

3. If all ρ_i are jump rules, then we use transformation (iii).

We now have another pre-model M′ with leaves γ_i, as well as the other leaves in L, if any. The maximum height of the traces rooted at the leaves γ_i is k. If any other leaves in L are rooted at traces of height $k + 1$, we repeat this process for all such leaves. Eventually, all members of L are rooted at traces of height $\leq k$, and then the induction hypothesis applies, and we can obtain a pre-model M′ such that all leaves of M′ are marked C. Q.E.D.

Theorem 4.4.19. *If A is* `BiInt`*-valid then* $\text{Prove}(\emptyset \Rightarrow A)$ *returns true.*

Proof. As is usual in semantic completeness proofs, we show the contrapositive: if $\text{Prove}(\emptyset \Rightarrow A)$ returns *false*, then A is not `BiInt`-valid.

Suppose that $\text{Prove}(\emptyset \Rightarrow A)$ returns *false*. Let T be the failed trace of Prove for $\emptyset \Rightarrow A$. Let M be a pre-model consisting of one I-node $\langle \emptyset, \{A\} \rangle$. Then by Lemma 4.4.18, there exists a pre-model rooted at $\langle \emptyset, \{A\} \rangle$ such that all leaves are complete. Then by Lemma 4.4.15, $\langle \emptyset, \{A\} \rangle$ has a counter-model, that is, there exists an $M = \langle W, \leq, V \rangle$ such that $w \dashv\!\Vdash A$ for some $w \in W$. Then A is not `BiInt`-valid. Q.E.D.

By Theorems 4.4.1 and 4.4.19, we have:

Theorem 4.4.20. *Any formula A is* $\mathbf{LBiInt_2}$*-derivable if and only if* $\text{Prove}(\emptyset \Rightarrow A)$ *returns true.*

Thus $\mathbf{LBiInt_2}$ gives us a decision procedure for `BiInt`. The fact that `BiInt` is decidable is already known, as we showed in Chapter 2.

A deep inference nested sequent calculus for bi-intuitionistic logic

In this chapter we present a deep inference calculus **DBiInt** for `BiInt`, and show that it is complete with respect to the shallow inference calculus $\mathbf{LBiInt_1}$ we presented in the previous chapter. First, in Section 5.1, we recall the syntax of our nested sequents and present our deep inference calculus **DBiInt**. In Section 5.2, we show that provability in **DBiInt** is equivalent to provability in $\mathbf{LBiInt_1}$, which is the central result of this chapter. The non-trivial part is showing that the residuation rules of $\mathbf{LBiInt_1}$ can be simulated by the propagation rules and deep inference of **DBiInt**. In Section 5.3, we give a simple restriction of **DBiInt** that allows terminating backward proof search.

Note. Some of the results of this chapter have been published in [98].

5.1 The sequent calculus DBiInt

Recall that a structure is defined by the following grammar, where A is a `BiInt` formula:

$$X := \emptyset \mid A \mid (X, X) \mid X \triangleright X.$$

If X and Y are structures, then $X \Rightarrow Y$ is a *nested shallow sequent* as defined previously, and $X \triangleright Y$ is a *nested deep sequent*. The definitions of polarities and contexts remain unchanged from the previous chapter.

We define the *immediate super-structure* of a context as: $\widehat{\Sigma[]} = X \triangleright Y$ such that $X \triangleright Y$ is a sub-structure of Σ and $X = [], X'$ for some structure X' or $Y = [], Y'$ for some structure Y'.

We define the *top-level* formulae of a structure as before:

$$\{|X|\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}.$$

For example, if $\Sigma[] = A, B \triangleright C, (D, (E \triangleright F) \triangleright [])$, then $\widehat{\Sigma[G]} = (D, (E \triangleright F) \triangleright G)$, and $\{|D, (E \triangleright F)|\} = \{D\}$.

While deep inference allows us to "zoom-in" to any sub-structure deep inside the nested sequent, the concept of an immediate super-structure acts the opposite way in that it allows us to "zoom-out" from a context to its immediate surrounding nested structure. This will be useful when we restrict our rules for terminating proof-search, allowing us to impose local checks on the rules.

The display property of pure display calculi is the ability to display/un-display a particular structure with respect to a top-level turnstile $\vdash$ (say) as the *whole* of the antecedent or succedent. For example, we have to display $V \triangleright W$ as the whole of the antecedent or succedent as $V \triangleright W \vdash Z$ or $Z \vdash V \triangleright W$. As mentioned previously in Chapter 4, our shallow nested sequent calculus $\mathbf{LBiInt_1}$ instead enables us to "zoom in" to $V \triangleright W$ in $X \Rightarrow Y$ by explicitly transforming the latter into $X', V \Rightarrow W, Y'$ so we can apply a rule to any top-level formula/structure of V or W. Our deep nested sequent calculus **DBiInt** is even more efficient since it allows us to "zoom in" to $V \triangleright W$ by treating it as the filler of a hole $\Sigma[V \triangleright W]$, requiring no explicit transformations.

Figure 5.1 gives the rules of our deep inference calculus **DBiInt**. Here the inference rules can be applied at any level of the nested sequent, indicated by the use of contexts. Notably, there are no residuation rules; indeed the main goal of this chapter is to show that the residuation rules of $\mathbf{LBiInt_1}$ can be simulated by deep inference and propagation rules in **DBiInt**. We write $\vdash_{\mathbf{DBiInt}} \Pi : X \triangleright Y$ to mean that there exists a **DBiInt**-derivation Π of the sequent $X \triangleright Y$.

We write $|\Pi|$ for the height of a derivation, i.e., the number of sequents on the longest branch, where Π is either an $\mathbf{LBiInt_1}$-derivation or a **DBiInt**-derivation.

5.1.1 Examples

We give two examples to illustrate the difference between shallow inference in $\mathbf{LBiInt_1}$ and deep inference in **DBiInt**.

Example 5.1.1. *The following is a derivation of Uustalu's formula [95] in* $\mathbf{LBiInt_1}$*:*

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{id}
\UnaryInfC{$p \Rightarrow q, p$}
\AxiomC{}
\RightLabel{id}
\UnaryInfC{$p, q \Rightarrow q$}
\RightLabel{$\mathrel{-\!\!<}_R$}
\BinaryInfC{$p \Rightarrow q, p \mathrel{-\!\!<} q$}
\RightLabel{$\triangleright_L$}
\UnaryInfC{$p \triangleright q \Rightarrow p \mathrel{-\!\!<} q$}
\RightLabel{w_L}
\UnaryInfC{$p \triangleright q, r \Rightarrow p \mathrel{-\!\!<} q$}
\AxiomC{}
\RightLabel{id}
\UnaryInfC{$(p \triangleright q), r \Rightarrow r$}
\RightLabel{$\wedge_R$}
\BinaryInfC{$(p \triangleright q), r \Rightarrow (p \mathrel{-\!\!<} q) \wedge r$}
\RightLabel{$\rightarrow_R$}
\UnaryInfC{$p \triangleright q \Rightarrow r \rightarrow ((p \mathrel{-\!\!<} q) \wedge r)$}
\RightLabel{s_L}
\UnaryInfC{$p \Rightarrow q, r \rightarrow ((p \mathrel{-\!\!<} q) \wedge r)$}
\end{prooftree}
$$

This example uses the rules $\triangleright_L$ and s_L to bring the required sub-structures to the top-level to apply the inference rules.

Example 5.1.2. *The following is a derivation of Uustalu's formula in* **DBiInt** *where we abbreviate $A = r \rightarrow ((p \mathrel{-\!\!<} q) \wedge r)$, $B = (p \mathrel{-\!\!<} q) \wedge r$ and $X = r \triangleright B, p \mathrel{-\!\!<} q$ to save space. For readability, we draw a box around the conclusion structure of the inference rule instance,*

Identity and logical constants:

$$\dfrac{}{\Sigma[X, A \triangleright A, Y]}\;id \qquad \dfrac{}{\Sigma^{-}[\bot]}\;\bot_L \qquad \dfrac{}{\Sigma^{+}[\top]}\;\top_R$$

Propagation rules:

$$\dfrac{\Sigma^{-}[A, (A, X \triangleright Y)]}{\Sigma^{-}[A, X \triangleright Y]}\;\triangleright_{L1} \qquad \dfrac{\Sigma^{+}[(X \triangleright Y, A), A]}{\Sigma^{+}[X \triangleright Y, A]}\;\triangleright_{R1}$$

$$\dfrac{\Sigma[A, X \triangleright (W, (A, Y \triangleright Z))]}{\Sigma[A, X \triangleright (W, (Y \triangleright Z))]}\;\triangleright_{L2} \qquad \dfrac{\Sigma[((X \triangleright Y, A), W) \triangleright Z, A]}{\Sigma[((X \triangleright Y), W) \triangleright Z, A]}\;\triangleright_{R2}$$

Logical rules:

$$\dfrac{\Sigma^{-}[A \wedge B, A, B]}{\Sigma^{-}[A \wedge B]}\;\wedge_L \qquad \dfrac{\Sigma^{+}[A \wedge B, A] \qquad \Sigma^{+}[A \wedge B, B]}{\Sigma^{+}[A \wedge B]}\;\wedge_R$$

$$\dfrac{\Sigma^{-}[A \vee B, A] \qquad \Sigma^{-}[A \vee B, B]}{\Sigma^{-}[A \vee B]}\;\vee_L \qquad \dfrac{\Sigma^{+}[A \vee B, A, B]}{\Sigma^{+}[A \vee B]}\;\vee_R$$

$$\dfrac{\Sigma^{-}[A \mathbin{-\!\!<} B, (A \triangleright B)]}{\Sigma^{-}[A \mathbin{-\!\!<} B]}\;{-\!\!<}_L \qquad \dfrac{\Sigma^{+}[A \to B, (A \triangleright B)]}{\Sigma^{+}[A \to B]}\;\to_R$$

$$\dfrac{\Sigma[X, A \to B \triangleright A, Y] \qquad \Sigma[X, A \to B, B \triangleright Y]}{\Sigma[X, A \to B \triangleright Y]}\;\to_L$$

$$\dfrac{\Sigma[X \triangleright Y, A \mathbin{-\!\!<} B, A] \qquad \Sigma[X, B \triangleright Y, A \mathbin{-\!\!<} B]}{\Sigma[X \triangleright Y, A \mathbin{-\!\!<} B]}\;{-\!\!<}_R$$

Figure 5.1: **DBiInt**: a deep inference nested sequent calculus for `BiInt`

unless it is the top-level structure:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{p \triangleright q, A, X, p \mathbin{-\!\!<} q, p}\;id \qquad \dfrac{}{p, q \triangleright q, A, X, p \mathbin{-\!\!<} q}\;id}{p \triangleright q, A, \boxed{(r \triangleright B, p \mathbin{-\!\!<} q)}, p \mathbin{-\!\!<} q}\;{-\!\!<}_R}{p \triangleright q, A, \boxed{(r \triangleright B, p \mathbin{-\!\!<} q)}}\;\triangleright_{R1} \qquad \dfrac{}{p \triangleright q, A, \boxed{(r \triangleright B, r)}}\;id}{\dfrac{p \triangleright q, A, \boxed{(r \triangleright (p \mathbin{-\!\!<} q) \wedge r)}}{p \triangleright q, r \to ((p \mathbin{-\!\!<} q) \wedge r)}\;\to_R}\;\wedge_R$$

This example uses deep inference to apply the inference rules at any level. The formula propagation rule $\triangleright_{R1}$ ensures that the required formula is propagated to the appropriate sub-structure.

5.2 Soundness and completeness of DBiInt

We now show that **DBiInt** is equivalent to the cut-free fragment of the sequent calculus $\mathbf{LBiInt_1}$ that we presented in Chapter 4.

5.2.1 Soundness of DBiInt

We show the soundness of **DBiInt** first, that is, that every rule of **DBiInt** can be derived in $\mathbf{LBiInt_1}$. This involves showing that the propagation rules of **DBiInt** can be derived in $\mathbf{LBiInt_1}$ using residuation. This is not a surprising result, since the residuation rules in display logics are used exactly for the purpose of displaying and un-displaying sub-sequents so that inference rules can be applied to them.

Theorem 5.2.1 (Soundness). *For any structures X and Y, if $\vdash_{\mathbf{DBiInt}} \Pi : X \triangleright Y$ then $\vdash_{\mathbf{LBiInt_1}}$ $\Pi' : X \Rightarrow Y$.*

Proof. We show that each deep inference rule ρ of **DBiInt** is derivable in $\mathbf{LBiInt_1}$. This is done by case analysis of the context $\Sigma[\,]$ in which the deep rule ρ applies. Note that if a deep inference rule ρ is applicable to $X \triangleright Y$, then the context $\Sigma[\,]$ in this case is either $[\,]$, a positive context or a negative context. In the first case, it is easy to show that each instance of ρ where $\Sigma[\,] = [\,]$ is derivable in the shallow system.

For the case where $\Sigma[\,]$ is either positive or negative, we use the display properties of $\mathbf{LBiInt_1}$. We show here the case where ρ is a rule with a single premise; the other cases are analogous. Suppose ρ is

$$\frac{\Sigma^+[U]}{\Sigma^+[V]}\,\rho$$

By the display properties of $\mathbf{LBiInt_1}$ (Lemmas 4.1.3 to 4.1.5), we only need to show that the following rules are derivable in the shallow system for some structure W':

$$\frac{W' \Rightarrow U}{W' \Rightarrow V} \qquad \frac{U \Rightarrow W'}{V \Rightarrow W'}$$

For example, to show soundness of $\triangleright_{L1}$ it is enough to show that the following are derivable:

$$\frac{W' \Rightarrow (A, (A, X \triangleright Y) \triangleright Z)}{W' \Rightarrow ((A, X \triangleright Y) \triangleright Z)} \qquad \frac{(A, (A, X \triangleright Y) \triangleright Z) \Rightarrow W'}{((A, X \triangleright Y) \triangleright Z) \Rightarrow W'}$$

Both reduce to showing that the following is derivable:

$$\frac{A, (A, X \triangleright Y) \triangleright Z}{(A, X \triangleright Y) \triangleright Z}$$

The following is the required derivation:

$$\cfrac{\cfrac{\cfrac{A, (A, X \triangleright Y) \Rightarrow Z}{A, A, X \Rightarrow Y, Z}\,s_L}{A, X \Rightarrow Y, Z}\,c_L}{(A, X \triangleright Y) \Rightarrow Z}\,\triangleright_L$$

Below are the other non-trivial cases (we give the **DBiInt** rule on the left and its derivation in $\mathbf{LBiInt_1}$ on the right):

$$\dfrac{Z \triangleright (X \triangleright Y, A), A}{Z \triangleright (X \triangleright Y, A)}\,\triangleright_{R1} \qquad \dfrac{\dfrac{\dfrac{Z \Rightarrow (X \triangleright Y, A), A}{Z, X \Rightarrow Y, A, A}\,s_R}{Z, X \Rightarrow Y, A}\,c_R}{Z \Rightarrow (X \triangleright Y, A)}\,\triangleright_R$$

$$\dfrac{A, X \triangleright (W, (A, Y \triangleright Z))}{A, X \triangleright (W, (Y \triangleright Z))}\,\triangleright_{L2} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A, X \Rightarrow W, (A, Y \triangleright Z)}{A, X \triangleright W \Rightarrow A, Y \triangleright Z}\,\triangleright_L}{(A, X \triangleright W), A, Y \Rightarrow Z}\,s_R}{(A, X \triangleright W), A \Rightarrow Y \triangleright Z}\,\triangleright_R}{A, X \triangleright W \Rightarrow A \triangleright (Y \triangleright Z)}\,\triangleright_R}{A, X \Rightarrow W, (A \triangleright (Y \triangleright Z))}\,s_L}{A, A, X \Rightarrow W, (Y \triangleright Z)}\,s_R}{A, X \Rightarrow W, (Y \triangleright Z)}\,c_L$$

$$\dfrac{((X \triangleright Y, A), W) \triangleright Z, A}{((X \triangleright Y), W) \triangleright Z, A}\,\triangleright_{R2} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(X \triangleright Y, A), W \Rightarrow Z, A}{X \triangleright Y, A \Rightarrow W \triangleright Z, A}\,\triangleright_R}{X \Rightarrow Y, A, (W \triangleright Z, A)}\,s_L}{X \triangleright Y \Rightarrow A, (W \triangleright Z, A)}\,\triangleright_L}{(X \triangleright Y) \triangleright A \Rightarrow W \triangleright Z, A}\,\triangleright_L}{((X \triangleright Y) \triangleright A), W \Rightarrow Z, A}\,s_R}{(X \triangleright Y), W \Rightarrow Z, A, A}\,s_L}{(X \triangleright Y), W \Rightarrow Z, A}\,c_R$$

Q.E.D.

5.2.2 Completeness of DBiInt

Our aim is to show that **DBiInt** is complete w.r.t. $\mathbf{LBiInt_1}$. But first we need some basic lemmas. Note that all the rules of **DBiInt** have been deliberately designed with backward proof search in mind, so it is not surprising that the proofs of the following three lemmas are so simple.

Lemma 5.2.2 (Admissibility of general weakening)**.** *For any context* Σ *and any structures* X *and* Y*: if* $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[X]$ *then* $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[X, Y]$ *such that* $|\Pi'| = |\Pi|$.

Proof. We give one case where Π ends with a propagation rule, and one case where Π ends with a logical rule; all other cases are analogous. In each of the following cases, we use induction on $|\Pi|$, and obtain Π_1' from Π_1 using the induction hypothesis.

$$\dfrac{\begin{matrix}\Pi_1\\ \Sigma^-[A, (A, X_1 \triangleright X_1)]\end{matrix}}{\Sigma^-[A, X_1 \triangleright X_2]}\,\triangleright_{L1} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ \Sigma^-[A, (A, X_1 \triangleright X_1), Y]\end{matrix}}{\Sigma^-[(A, X_1 \triangleright X_2), Y]}\,\triangleright_{L1}$$

$$\dfrac{\begin{matrix}\Pi_1\\ \Sigma^+[A \to B, (A \triangleright B)]\end{matrix}}{\Sigma^+[A \to B]}\,\to_R \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ \Sigma^+[A \to B, (A \triangleright B), Y]\end{matrix}}{\Sigma^+[(A \to B), Y]}\,\to_R$$

Q.E.D.

Lemma 5.2.3 (Admissibility of formula contraction). *For any context* Σ *and any structure* X *and formula* A*: if* $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[X, A, A]$ *then* $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[X, A]$ *such that* $|\Pi'| = |\Pi|$.

Proof. We give two cases where Π ends with a propagation rule, and two cases where Π ends with a logical rule; all other cases are analogous. In each of the following cases, we use a induction on $|\Pi|$, and obtain Π_1' from Π_1 using the induction hypothesis.

$$\dfrac{\overset{\Pi_1}{\Sigma^-[A, (A, A, X_1 \triangleright X_1)]}}{\Sigma^-[A, A, X_1 \triangleright X_2]}\,\triangleright_{L1} \quad\leadsto\quad \dfrac{\overset{\Pi_1'}{\Sigma^-[A, (A, X_1 \triangleright X_1)]}}{\Sigma^-[(A, X_1 \triangleright X_2)]}\,\triangleright_{L1}$$

$$\dfrac{\overset{\Pi_1}{\Sigma[((X \triangleright Y, A), W) \triangleright Z, A, A]}}{\Sigma[((X \triangleright Y), W) \triangleright Z, A, A]}\,\triangleright_{R2} \quad\leadsto\quad \dfrac{\overset{\Pi_1'}{\Sigma[((X \triangleright Y, A), W) \triangleright Z, A]}}{\Sigma[((X \triangleright Y), W) \triangleright Z, A]}\,\triangleright_{R2}$$

$$\dfrac{\overset{\Pi_1}{\Sigma^-[A \prec B, A \prec B, (A \triangleright B)]}}{\Sigma^-[A \prec B, A \prec B]}\,\prec_L \quad\leadsto\quad \dfrac{\overset{\Pi_1'}{\Sigma^-[A \prec B, (A \triangleright B)]}}{\Sigma^-[A \prec B]}\,\prec_L$$

$$\dfrac{\overset{\Pi_1}{\Sigma^+[A \vee B, A \vee B, A, B]}}{\Sigma^+[A \vee B, A \vee B]}\,\vee_R \quad\leadsto\quad \dfrac{\overset{\Pi_1'}{\Sigma^+[A \vee B, A, B]}}{\Sigma^+[A \vee B]}\,\vee_R$$

Q.E.D.

Invertibility of our rules follows immediately, since or each of our rules, the premise is a superset of the conclusion, and weakening is height-preserving.

Lemma 5.2.4 (Invertibility). *All* **DBiInt** *rules are invertible: if the conclusion is derivable, then each premise is derivable.*

We now show that the residuation rules of $\mathbf{LBiInt_1}$ are admissible in **DBiInt**; that is, they can be simulated by the propagation rules of **DBiInt**. Actually, what we show is next is a stronger result: the admissibility of "deep" versions of these rules, which is important for later showing the completeness of our proof search calculus in Section 5.3.

Lemmas 5.2.5 to 5.2.8 are proved by structural induction on $\Sigma[]$, and a sub-induction on $|\Pi|$. In each of the derivations, a dashed inference line means that the conclusion is obtained from the premise using the respective Lemma.

Lemma 5.2.5 (Admissibility of s_L). *For any context* $\Sigma[]$*, if* $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[(X \triangleright Y), Z \triangleright W]$ *then* $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[X, Z \triangleright Y, W]$ *such that* $|\Pi'| \leq |\Pi|$.

Proof.

- First we show the base case when $\Sigma[] = []$. We use a sub-induction on the height of the derivation Π, and obtain Π_1' (resp. Π_2') from Π_1 (resp. Π_2) using the sub-induction hypothesis.

  - Cases when Π ends with a propagation rule that moves formulae within the structures X and Y:

$$\dfrac{\begin{matrix}\Pi_1\\ (A,(A,X_1 \triangleright X_2) \triangleright Y), Z \triangleright W\end{matrix}}{((A,X_1 \triangleright X_2) \triangleright Y), Z \triangleright W}\,\triangleright_{L1} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ A,(A,X_1 \triangleright X_2), Z \triangleright Y, W\end{matrix}}{(A,X_1 \triangleright X_2), Z \triangleright Y, W}\,\triangleright_{L1}$$

$$\dfrac{\begin{matrix}\Pi_1\\ (X \triangleright (Y_1 \triangleright Y_2, A), A), Z \triangleright W\end{matrix}}{(X \triangleright (Y_1 \triangleright Y_2, A)), Z \triangleright W}\,\triangleright_{R1} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ X, Z \triangleright (Y_1 \triangleright Y_2, A), A, W\end{matrix}}{X, Z \triangleright (Y_1 \triangleright Y_2, A), W}\,\triangleright_{R1}$$

  - Cases when Π ends with a propagation rule that moves formulae between the structures X, Y, Z, W:

$$\dfrac{\begin{matrix}\Pi_1\\ (A, X \triangleright (A, Y_1 \triangleright Y_2)), Z \triangleright W\end{matrix}}{(A, X \triangleright (Y_1 \triangleright Y_2)), Z \triangleright W}\,\triangleright_{L2} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ A, X, Z \triangleright (A, Y_1 \triangleright Y_2), W\end{matrix}}{A, X, Z \triangleright (Y_1 \triangleright Y_2), W}\,\triangleright_{L2}$$

$$\dfrac{\begin{matrix}\Pi_1\\ (X \triangleright Y, A), Z \triangleright W_1, A\end{matrix}}{(X \triangleright Y), Z \triangleright W_1, A}\,\triangleright_{R2} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ X, Z \triangleright Y, W_1, A, A\end{matrix}}{X, Z \triangleright Y, W_1, A}\ \text{Lemma 5.2.3}$$

  - Case when Π ends with the logical rule $\to_L$ where the principal formula is in X:

$$\dfrac{\begin{matrix}\Pi_1\\ (X_1, A \to B \triangleright A, Y), Z \triangleright W\end{matrix} \qquad \begin{matrix}\Pi_2\\ (X_1, A \to B, B \triangleright Y), Z \triangleright W\end{matrix}}{(X_1, A \to B \triangleright Y), Z \triangleright W}\,\to_L \quad \rightsquigarrow$$

$$\dfrac{\begin{matrix}\Pi_1'\\ X_1, A \to B, Z \triangleright A, Y, W\end{matrix} \qquad \begin{matrix}\Pi_2'\\ X_1, A \to B, B, Z \triangleright Y, W\end{matrix}}{X_1, A \to B, Z \triangleright Y, W}\,\to_L$$

  - Case when Π ends with the logical rule $\prec_R$ where the principal formula is in Y:

$$\dfrac{\begin{matrix}\Pi_1\\ (X \triangleright Y_1, A \prec B, A), Z \triangleright W\end{matrix} \qquad \begin{matrix}\Pi_2\\ (X, B \triangleright Y_1, A \prec B), Z \triangleright W\end{matrix}}{(X \triangleright Y_1, A \prec B), Z \triangleright W}\,\prec_L \quad \rightsquigarrow$$

$$\dfrac{\begin{matrix}\Pi_1'\\ X, Z \triangleright Y_1, A \prec B, A, W\end{matrix} \qquad \begin{matrix}\Pi_2'\\ X, B, Z \triangleright Y_1, A \prec B, W\end{matrix}}{X, Z \triangleright Y_1, A \prec B, W}\,\prec_L$$

  - The cases involving other rules follow immediately from the sub-induction hypothesis, since they do not move formulae across $\triangleright$-structures.

- For the inductive cases, we have either (1) $\Sigma[] = \Sigma_1[([], U) \rhd V]$ or (2) $\Sigma[] = \Sigma_1[U \rhd (V, [])]$ for some (possibly empty) structures U and V and some context $\Sigma_1[]$.

  We first show case (1) when $\Sigma[] = \Sigma_1[([], U) \rhd V]$. We consider sub-cases when a formula is either propagated out from the context $\Sigma[]$ to U, or from V into the context $\Sigma[]$. In each case below, we obtain Π_1' from Π_1 using the induction hypothesis, applied to the context $\Sigma_1[]$.

  - Case when Π ends with a propagation rule that moves a formula out from $\Sigma[]$ to U:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma_1[(((X \rhd Y), Z_1, A \rhd W), A, U) \rhd V]\end{array}}{\Sigma_1[(((X \rhd Y), Z_1, A \rhd W), U) \rhd V]}\,\rhd_{L1} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma_1[((X, Z_1, A \rhd Y, W), A, U) \rhd V]\end{array}}{\Sigma_1[((X, Z_1, A \rhd Y, W), U) \rhd V]}\,\rhd_{L1}$$

  - Case when Π ends with a propagation rule that moves a formula from V into $\Sigma[]$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma_1[(((X \rhd Y), Z \rhd W, A), U) \rhd V_1, A]\end{array}}{\Sigma_1[(((X \rhd Y), Z \rhd W), U) \rhd V_1, A]}\,\rhd_{R2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma_1[(X, Z \rhd Y, W, A), U \rhd V_1, A]\end{array}}{\Sigma_1[(X, Z \rhd Y, W), U \rhd V_1, A]}\,\rhd_{R2}$$

  - The cases when formulae are propagated within the context can be proven identically to the case when $\Sigma[] = []$.

  We now show case (2) when $\Sigma[] = \Sigma_1[U \rhd (V, [])]$. We consider sub-cases when a formula is either propagated out from the context $\Sigma[]$ to V, or from U into the context $\Sigma[]$. In each case below, we obtain Π_1' from Π_1 using the induction hypothesis, applied to the context $\Sigma_1[]$.

  - Case when Π ends with a propagation rule that moves a formula from $\Sigma[]$ into V:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma_1[U \rhd (V, ((X \rhd Y), Z \rhd W_1, A), A)]\end{array}}{\Sigma_1[U \rhd (V, ((X \rhd Y), Z \rhd W_1, A))]}\,\rhd_{R1} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma_1[U \rhd (V, (X, Z \rhd Y, W_1, A), A)]\end{array}}{\Sigma_1[U \rhd (V, (X, Z \rhd Y, W_1, A))]}\,\rhd_{R1}$$

  - Case when Π ends with a propagation rule that moves a formula from U into $\Sigma[]$:

$$\dfrac{\dfrac{\Pi_1}{\Sigma_1[U_1, A \triangleright (V, (A, (X \triangleright Y), Z \triangleright W))]}}{\Sigma_1[U_1, A \triangleright (V, ((X \triangleright Y), Z \triangleright W))]} \triangleright_{L2} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\Pi_1'}{\Sigma_1[U_1, A \triangleright (V, (A, X, Z \triangleright Y, W))]}}{\Sigma_1[U_1, A \triangleright (V, (X, Z \triangleright Y, W))]} \triangleright_{L2}$$

Q.E.D.

Lemma 5.2.6 (Admissibility of s_R). *For any context* $\Sigma[]$, *if* $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[X \triangleright Y, (Z \triangleright W)]$ *then* $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[X, Z \triangleright Y, W]$ *such that* $|\Pi'| \leq |\Pi|$.

Proof. Symmetric to the proof of Lemma 5.2.5; detailed in Section B.1. Q.E.D.

While the rules s_L and s_R are deeply admissible in any context, the rule $\triangleright_L$ is only admissible at the top level and in a negative context, and the rule $\triangleright_L$ is only admissible at the top level and in a positive context. Note also that unlike s_L and s_R, the rules $\triangleright_L$ and $\triangleright_R$ are not height-preserving admissible.

Lemma 5.2.7 (Admissibility of $\triangleright_L$). *For any context* $\Sigma[]$ *such that either* $\Sigma[] = []$ *or* $\Sigma[]$ *is a negative context, if* $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[X \triangleright Y, Z]$ *then* $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[(X \triangleright Y) \triangleright Z]$.

Proof.

- First we show the base case when $\Sigma[] = []$. We use a sub-induction on the height of the derivation Π, and obtain Π_1' (resp. Π_2') from Π_1 (resp. Π_2) using the sub-induction hypothesis.
  - Cases when Π ends with a propagation rule that moves formulae between the structures X and Y:

$$\dfrac{\dfrac{\Pi_1}{A, X_1 \triangleright (A, Y_1 \triangleright Y_2), Z}}{A, X_1 \triangleright (Y_1 \triangleright Y_2), Z} \triangleright_{L2} \quad \rightsquigarrow \quad \dfrac{\dfrac{\Pi_1'}{(A, X_1 \triangleright (A, Y_1 \triangleright Y_2)) \triangleright Z}}{(A, X_1 \triangleright (Y_1 \triangleright Y_2)) \triangleright Z} \triangleright_{L2}$$

$$\dfrac{\dfrac{\Pi_1}{(X_1 \triangleright X_2, A) \triangleright Y_1, A, Z}}{(X_1 \triangleright X_2) \triangleright Y_1, A, Z} \triangleright_{R2} \quad \rightsquigarrow \quad \dfrac{\dfrac{\Pi_1'}{((X_1 \triangleright X_2, A) \triangleright Y_1, A) \triangleright Z}}{((X_1 \triangleright X_2) \triangleright Y_1, A) \triangleright Z} \triangleright_{R2}$$

  - Cases when Π ends with a propagation rule that moves formulae between the structures X and Z:

$$\dfrac{\begin{array}{c}\Pi_1\\ A, X_1 \triangleright Y, (A, Z_1 \triangleright Z_2)\end{array}}{A, X_1 \triangleright Y, (Z_1 \triangleright Z_2)}\triangleright_{L2} \quad\rightsquigarrow\quad \dfrac{\dfrac{\begin{array}{c}\Pi_1'\\ (A, X_1 \triangleright Y) \triangleright (A, Z_1 \triangleright Z_2)\\ \hdashline A, (A, X_1 \triangleright Y) \triangleright (A, Z_1 \triangleright Z_2)\end{array}\text{Lm. 5.2.2}}{A, (A, X_1 \triangleright Y) \triangleright (Z_1 \triangleright Z_2)}\triangleright_{L2}}{(A, X_1 \triangleright Y) \triangleright (Z_1 \triangleright Z_2)}\triangleright_{L1}$$

$$\dfrac{\begin{array}{c}\Pi_1\\ (X_1 \triangleright X_2, A) \triangleright Y, Z_1, A\end{array}}{(X_1 \triangleright X_2) \triangleright Y, Z_1, A}\triangleright_{R2} \quad\rightsquigarrow\quad \dfrac{\dfrac{\begin{array}{c}\Pi_1'\\ ((X_1 \triangleright X_2, A) \triangleright Y) \triangleright Z_1, A\\ \hdashline ((X_1 \triangleright X_2, A) \triangleright Y, A) \triangleright Z_1, A\end{array}\text{Lm. 5.2.2}}{((X_1 \triangleright X_2) \triangleright Y, A) \triangleright Z_1, A}\triangleright_{R2}}{((X_1 \triangleright X_2) \triangleright Y) \triangleright Z_1, A}\triangleright_{R2}$$

– Cases when Π ends with a propagation rule that moves formulae within the structures X and Y:

$$\dfrac{\begin{array}{c}\Pi_1\\ A, (A, X_1 \triangleright X_2) \triangleright Y, Z\end{array}}{(A, X_1 \triangleright X_2) \triangleright Y, Z}\triangleright_{L1} \quad\rightsquigarrow\quad \dfrac{\begin{array}{c}\Pi_1'\\ (A, (A, X_1 \triangleright X_2) \triangleright Y) \triangleright Z\end{array}}{((A, X_1 \triangleright X_2) \triangleright Y) \triangleright Z}\triangleright_{L1}$$

$$\dfrac{\begin{array}{c}\Pi_1\\ X \triangleright (Y_1 \triangleright Y_2, A), A, Z\end{array}}{X \triangleright (Y_1 \triangleright Y_2, A), Z}\triangleright_{R1} \quad\rightsquigarrow\quad \dfrac{\begin{array}{c}\Pi_1'\\ (X \triangleright (Y_1 \triangleright Y_2, A), A) \triangleright Z\end{array}}{(X \triangleright (Y_1 \triangleright Y_2, A)) \triangleright Z}\triangleright_{R1}$$

– Case when Π ends with a $\to_L$ rule where the principal formula is in X:

$$\dfrac{\begin{array}{c}\Pi_1\\ X_1, A \to B \triangleright A, Y, Z\end{array} \qquad \begin{array}{c}\Pi_2\\ X_1, A \to B, B \triangleright Y, Z\end{array}}{X_1, A \to B \triangleright Y, Z}\to_L \quad\rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ (X_1, A \to B \triangleright A, Y) \triangleright Z\end{array} \qquad \begin{array}{c}\Pi_2'\\ (X_1, A \to B, B \triangleright Y) \triangleright Z\end{array}}{(X_1, A \to B \triangleright Y) \triangleright Z}\to_L$$

– Case when Π ends with a $-\!\!<_R$ rule where the principal formula is in Y:

$$\dfrac{\begin{array}{c}\Pi_1\\ X \triangleright Y_1, A -\!\!< B, A, Z\end{array} \qquad \begin{array}{c}\Pi_2\\ X, B \triangleright Y_1, A -\!\!< B, Z\end{array}}{X \triangleright Y_1, A -\!\!< B, Z}-\!\!<_R \quad\rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ (X \triangleright Y_1, A -\!\!< B, A) \triangleright Z\end{array} \qquad \begin{array}{c}\Pi_2'\\ (X, B \triangleright Y_1, A -\!\!< B) \triangleright Z\end{array}}{(X \triangleright Y_1, A -\!\!< B) \triangleright Z}-\!\!<_R$$

– Case when Π ends with a $-\!\!<_R$ rule where the principal formula is in Z:

$$\dfrac{\overset{\Pi_1}{X \rhd Y, Z_1, A -\!\!< B, A} \qquad \overset{\Pi_2}{X, B \rhd Y, Z_1, A -\!\!< B}}{X \rhd Y, Z_1, A -\!\!< B} -\!\!<_R \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\overset{\Pi'_1}{(X \rhd Y, A -\!\!< B, A) \rhd Z_1} \qquad \overset{\Pi'_2}{(X, B \rhd Y, A -\!\!< B) \rhd Z_1}}{(X \rhd Y, A -\!\!< B) \rhd Z_1} -\!\!<_R}{(X \rhd Y, A -\!\!< B) \rhd Z_1, A -\!\!< B} \text{Lemma 5.2.2}}{(X \rhd Y) \rhd Z_1, A -\!\!< B} \rhd_{R2}$$

– The cases involving other rules follow immediately from the induction hypothesis, since they do not move formulae across $\rhd$-structures.

- For the inductive case, we have $\Sigma[] = \Sigma_1[([], U) \rhd V]$ for some (possibly empty) structures U and V and some context $\Sigma_1[]$. We now consider sub-cases when a formula is either propagated out from the context $\Sigma[]$ to U, or from V into the context $\Sigma[]$. In each case below, we obtain Π'_1 from Π_1 using the induction hypothesis, applied to the context $\Sigma_1[]$.

  – Case when Π ends with a propagation rule that moves a formula out from $\Sigma[]$ to U:

$$\dfrac{\overset{\Pi_1}{\Sigma_1[((X_1, A \rhd Y, Z), A, U) \rhd V]}}{\Sigma_1[((X_1, A \rhd Y, Z), U) \rhd V]} \rhd_{L1} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\overset{\Pi'_1}{\Sigma_1[(((X_1, A \rhd Y) \rhd Z), A, U) \rhd V]}}{\Sigma_1[((A, (X_1, A \rhd Y) \rhd Z), A, U) \rhd V]} \text{Lemma 5.2.2}}{\Sigma_1[((A, (X_1, A \rhd Y) \rhd Z), U) \rhd V]} \rhd_{L1}}{\Sigma_1[(((X_1, A \rhd Y) \rhd Z), U) \rhd V]} \rhd_{L1}$$

  – Case when Π ends with a propagation rule that moves a formula from V into $\Sigma[]$:

$$\dfrac{\overset{\Pi_1}{\Sigma_1[((X \rhd Y, Z, A), U) \rhd V_1, A]}}{\Sigma_1[((X \rhd Y, Z), U) \rhd V_1, A]} \rhd_{R2} \quad \rightsquigarrow$$

$$\dfrac{\overset{\Pi'_1}{\Sigma_1[(((X \rhd Y) \rhd Z, A), U) \rhd V_1, A]}}{\Sigma_1[(((X \rhd Y) \rhd Z), U) \rhd V_1, A]} \rhd_{R2}$$

  – The cases when formulae are propagated within the context can be proven identically to the case when $\Sigma[] = []$.

Q.E.D.

Lemma 5.2.8 (Admissibility of $\rhd_R$). *For any context $\Sigma[]$ such that either $\Sigma[] = []$ or $\Sigma[]$ is a positive context, if $\vdash_{\mathbf{DBiInt}} \Pi : X, Y \rhd Z$ then $\vdash_{\mathbf{DBiInt}} \Pi' : X \rhd (Y \rhd Z)$.*

Proof. Symmetric to the proof of Lemma 5.2.7; detailed in Section B.1. Q.E.D.

In order to show the admissibility of general contraction, we first need to show a distribution lemma.

Lemma 5.2.9 (Distribution lemma)**.** *For any context* $\Sigma[]$ *and for any structures* X, Y, Z, W*: if* $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[(X \rhd Y), (Z \rhd W)]$ *then* $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[(X, Z \rhd Y, W)]$ *such that* $|\Pi'| = |\Pi|$.

Proof. By induction on the height of Π; we obtain Π_1' from Π_1 using the induction hypothesis. As in the previous lemmas, the non-trivial cases are those where Π ends with a propagation rule applied to the structures X, Z, Y, W.

- Case when Π ends with a propagation rule that moves a formula from X to Y:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma[(X_1, A \rhd Y_1, (A, Y_2 \rhd Y_2)), (Z \rhd W)]\end{array}}{\Sigma[(X_1, A \rhd Y_1, (Y_2 \rhd Y_2)), (Z \rhd W)]}\rhd_{L2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma[(X_1, A, Z \rhd Y_1, (A, Y_2 \rhd Y_2), W)]\end{array}}{\Sigma[(X_1, A, Z \rhd Y_1, (Y_2 \rhd Y_2), W)]}\rhd_{L2}$$

- Case when Π ends with a propagation rule that moves a formula from Y to X:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma[((X_1 \rhd X_2, A), X_3 \rhd Y_1, A), (Z \rhd W)]\end{array}}{\Sigma[((X_1 \rhd X_2), X_3 \rhd Y_1, A), (Z \rhd W)]}\rhd_{R2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma[((X_1 \rhd X_2, A), X_3, Z \rhd Y_1, A, W)]\end{array}}{\Sigma[((X_1 \rhd X_2), X_3, Z \rhd Y_1, A, W)]}\rhd_{R2}$$

- Case when Π ends with a propagation rule that moves a formula out of Y and $\Sigma[]$ is a positive context:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma^+[(X \rhd Y_1, A), A, (Z \rhd W)]\end{array}}{\Sigma^+[(X \rhd Y_1, A), (Z \rhd W)]}\rhd_{R1} \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma^+[(X, Z \rhd Y_1, A, W), A]\end{array}}{\Sigma^+[(X, Z \rhd Y_1, A, W)]}\rhd_{R1}$$

- Case when Π ends with a propagation rule that moves a formula out of X and $\Sigma[]$ is a negative context:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma^-[A, (A, X_1 \rhd Y), (Z \rhd W)]\end{array}}{\Sigma^-[(A, X_1 \rhd Y), (Z \rhd W)]}\rhd_{L1} \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma^-[A, (A, X_1, Z \rhd Y, W)]\end{array}}{\Sigma^-[(A, X_1, Z \rhd Y, W)]}\rhd_{L1}$$

- Case when Π ends with a propagation rule that moves a formula into X from outside the context $\Sigma[]$:

$$\dfrac{\overset{\Pi_1}{\Sigma_1[U_1, A \triangleright (A, X \triangleright Y), (Z \triangleright W), U_2]}}{\Sigma_1[U_1, A \triangleright (X \triangleright Y), (Z \triangleright W), U_2]} \triangleright_{L2} \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{\Sigma_1[U_1, A \triangleright (A, X, Z \triangleright Y, W), U_2]}}{\Sigma_1[U_1, A \triangleright (X, Z \triangleright Y, W), U_2]} \triangleright_{L2}$$

- Case when Π ends with a propagation rule that moves a formula into Y from outside the context $\Sigma[]$:

$$\dfrac{\overset{\Pi_1}{\Sigma_1[U_1, (X \triangleright Y, A), (Z \triangleright W) \triangleright A, U_2]}}{\Sigma_1[U_1, (X \triangleright Y), (Z \triangleright W) \triangleright A, U_2]} \triangleright_{R2} \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{\Sigma_1[U_1, (X, Z \triangleright Y, W, A) \triangleright A, U_2]}}{\Sigma_1[U_1, (X, Z \triangleright Y, W) \triangleright A, U_2]} \triangleright_{R2}$$

Q.E.D.

Lemma 5.2.10 (Admissibility of general contraction). *For any context $\Sigma[]$ and for any structures X and Y: if $\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[X, Y, Y]$ then $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[X, Y]$ such that $|\Pi'| = |\Pi|$.*

Proof. By induction on the size of Y; the base case is proved by Lemma 5.2.3. The non-trivial case is when $Y = (Y_1 \triangleright Y_2)$. We use the distribution lemma to show how this can be reduced to contractions on Y_1 and Y_2, which are admissible by the induction hypothesis. A dashed inference line means that the conclusion is obtained from the premise using the respective Lemma or the induction hypothesis. Suppose we have Y in a negative context, the other case is symmetric:

$$\dfrac{\dfrac{\Sigma[(Y_1 \triangleright Y_2), (Y_1 \triangleright Y_2) \triangleright Z]}{\Sigma[(Y_1, Y_1 \triangleright Y_2, Y_2) \triangleright Z]} \text{ Lemma 5.2.9}}{\Sigma[(Y_1 \triangleright Y_2) \triangleright Z]} \text{ IH}$$

Q.E.D.

Having showed the admissibility of all structural rules of **LBiInt** in **DBiInt**, completeness is easy:

Theorem 5.2.11 (Completeness). *For any structures X and Y, if $\vdash_{\mathbf{LBiInt_1}} \Pi : X \Rightarrow Y$ then $\vdash_{\mathbf{DBiInt}} \Pi' : X \triangleright Y$.*

Proof. By induction on $|\Pi|$, where we obtain Π_1' (Π_2') from Π_1 (Π_2) using the induction hypothesis. A dashed inference line means that the conclusion is obtained from the premise using the respective Lemma. Because **LBiInt₁** has cut-elimination (Theorem 4.2.7), we only need to consider rules other than cut:

- Case when Π ends with w_L:

$$\cfrac{\begin{array}{c}\Pi_1\\ X \Rightarrow Y\end{array}}{X, A \Rightarrow Y}\, w_L \qquad \rightsquigarrow \qquad \begin{array}{c}\Pi_1'\\ X \rhd Y\\ \hdashline X, A \rhd Y\end{array}\ \text{Lemma 5.2.2}$$

- Case when Π ends with w_R: analogous to the case for w_L.

- Case when Π ends with c_L:

$$\cfrac{\begin{array}{c}\Pi_1\\ X, A, A \Rightarrow Y\end{array}}{X, A \Rightarrow Y}\, w_L \qquad \rightsquigarrow \qquad \begin{array}{c}\Pi_1'\\ X, A, A \rhd Y\\ \hdashline X, A \rhd Y\end{array}\ \text{Lemma 5.2.3}$$

- Case when Π ends with c_R: analogous to the case for c_L.

- Case when Π ends with s_L:

$$\cfrac{\begin{array}{c}\Pi_1\\ (X_1 \rhd Y_1), X_2 \Rightarrow Y_2\end{array}}{X_1, X_2 \Rightarrow Y_1, Y_2}\, s_L \qquad \rightsquigarrow \qquad \begin{array}{c}\Pi_1'\\ (X_1 \rhd Y_1), X_2 \rhd Y_2\\ \hdashline X_1, X_2 \rhd Y_1, Y_2\end{array}\ \text{Lemma 5.2.5}$$

- Case when Π ends with s_R: analogous to the case for s_L, using Lemma 5.2.6 instead.

- Case when Π ends with $\rhd_L$:

$$\cfrac{\begin{array}{c}\Pi_1\\ X_2 \Rightarrow Y_2, Y_1\end{array}}{(X_2 \rhd Y_2) \Rightarrow Y_1}\, \rhd_L \qquad \rightsquigarrow \qquad \begin{array}{c}\Pi_1'\\ X_2 \rhd Y_2, Y_1\\ \hdashline (X_2 \rhd Y_2) \rhd Y_1\end{array}\ \text{Lemma 5.2.7}$$

- Case when Π ends with $\rhd_R$: analogous to the case for $\rhd_L$, using Lemma 5.2.8 instead.

- Case when Π ends with $\to_L$:

$$\cfrac{\begin{array}{c}\Pi_1\\ X \Rightarrow A, Y\end{array} \qquad \begin{array}{c}\Pi_2\\ X, B \Rightarrow Y\end{array}}{X, A \to B \Rightarrow Y}\, \to_L \ \rightsquigarrow$$

$$\cfrac{\begin{array}{c}\Pi_1'\\ X \rhd A, Y\\ \hdashline X, A \to B \rhd A, Y\end{array}\ \text{Lemma 5.2.2} \qquad \begin{array}{c}\Pi_2'\\ X, B \Rightarrow Y\\ \hdashline X, A \to B, B \rhd Y\end{array}\ \text{Lemma 5.2.2}}{X, A \to B \rhd Y}\, \to_L$$

- Cases when Π ends with $-\!\!<_R$ and all rules for $\vee$, $\wedge$: analogous to the case for $\to_L$.

- Case when Π ends with $-\!\!\!<_L$:

$$\dfrac{\begin{matrix}\Pi_1\\ A \Rightarrow B, Y\end{matrix}}{X, A -\!\!\!< B \Rightarrow Y}\,-\!\!\!<_L \quad\rightsquigarrow\quad \dfrac{\dfrac{\dfrac{\begin{matrix}\Pi_1'\\ A \rhd B, Y\end{matrix}}{(A \rhd B) \rhd Y}\text{ Lemma 5.2.7}}{X, A -\!\!\!< B, (A \rhd B) \rhd Y}\text{ Lemma 5.2.2}}{X, A -\!\!\!< B \rhd Y}\,-\!\!\!<_L$$

- Case when Π ends with $\rightarrow_R$: analogous to the case for $-\!\!\!<_R$, using Lemma 5.2.8 instead.

Q.E.D.

Theorem 5.2.12. *For any structures X and Y, $\vdash_{\mathbf{LBiInt_1}} \Pi : X \Rightarrow Y$ if and only if $\vdash_{\mathbf{DBiInt}}$ $\Pi' : X \rhd Y$.*

Proof. By Theorems 5.2.1 and 5.2.11. Q.E.D.

5.3 Proof search

Naive proof search in **DBiInt** does not terminate, as illustrated by Example 5.3.1.

Example 5.3.1. *Consider the following proof attempt fragment, where $X = (A \rightarrow B) \rightarrow C, (D \rightarrow E) \rightarrow F$ and we only show the left premise of each $\rightarrow_L$ rule instance:*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{X \rhd G, A \rightarrow B, (X, A \rhd B, D \rightarrow E, (X, A, D \rhd E, A \rightarrow B, (A \rhd B)))}\rightarrow_R}{X \rhd G, A \rightarrow B, (X, A \rhd B, D \rightarrow E, \boxed{(X, A, D \rhd E, A \rightarrow B)})}\rightarrow_L}{X \rhd G, A \rightarrow B, \boxed{(X, A \rhd B, D \rightarrow E, (X, A, D \rhd E))}}2 \times \rhd_{L2}}{X \rhd G, A \rightarrow B, \boxed{(X, A \rhd B, D \rightarrow E, (D \rhd E))}}\rightarrow_R}{X \rhd G, A \rightarrow B, \boxed{(X, A \rhd B, D \rightarrow E)}}\rightarrow_L}{X \rhd G, A \rightarrow B, \boxed{(X, A \rhd B)}}2 \times \rhd_{L2}}{X \rhd G, A \rightarrow B, (A \rhd B)}\rightarrow_R}{X \rhd G, A \rightarrow B}\rightarrow_L}{(A \rightarrow B) \rightarrow C, (D \rightarrow E) \rightarrow F \rhd G}$$

In the example above, there is an interaction between the $\rightarrow_R$, $\rhd_{L2}$ and $\rightarrow_L$ rules that causes non-termination, even for the intuitionistic fragment of the logic. This well-known problem occurs in traditional sequent calculi as well, and it is caused by the implicit contraction in the $\rightarrow_L$ rule. For intuitionistic logic, this problem has been addressed by contraction-free calculi [36] and history-based loop-checks [63]. However, these methods are less suitable for `BiInt` where the interaction between $\rightarrow$ and $-\!\!\!<$ formulae needs to be considered: recall our discussion in Section 3.5.

Here we address termination using a saturation process and two derived rules that speed up proof search. The approach is similar to our work on **LBiInt$_1$** in Chapter 4, but here we apply it to deep inference and contexts instead of top-level sequents only.

Let $\mathrel{-\!\!<}_{L1}$ and $\rightarrow_{R1}$ denote two rules derived as below, where a dashed inference line means the conclusion is derived from the premise using Lemma 5.2.2. We show each rule on the left and its derivation on the right:

$$\dfrac{\Sigma^-[A, A \mathrel{-\!\!<} B]}{\Sigma^-[A \mathrel{-\!\!<} B]}\ \mathrel{-\!\!<}_{L1} \qquad\qquad \dfrac{\dfrac{\dfrac{\Sigma^-[A, A \mathrel{-\!\!<} B]}{\Sigma^-[A \mathrel{-\!\!<} B, A, (A \triangleright B)]}\ \text{Lemma 5.2.2}}{\Sigma^-[A \mathrel{-\!\!<} B, (A \triangleright B)]}\ \triangleright_{L1}}{\Sigma^-[A \mathrel{-\!\!<} B]}\ \mathrel{-\!\!<}_{L}$$

$$\dfrac{\Sigma^+[A \rightarrow B, B]}{\Sigma^+[A \rightarrow B]}\ \rightarrow_{R1} \qquad\qquad \dfrac{\dfrac{\dfrac{\Sigma^+[A \rightarrow B, B]}{\Sigma^+[A \rightarrow B, (A \triangleright B), B]}\ \text{Lemma 5.2.2}}{\Sigma^+[A \rightarrow B, (A \triangleright B)]}\ \triangleright_{R1}}{\Sigma^+[A \rightarrow B]}\ \rightarrow_{R}$$

Definition 5.3.2. *Let $\Sigma[Z]$ be any sequent. Then let $X \triangleright Y = \overbrace{\Sigma[Z]}$. We say that $\Sigma[Z]$ is* saturated *(w.r.t. $\Sigma[]$ and Z) iff all the following conditions are met:*

1. $\{\!|X|\!\} \cap \{\!|Y|\!\} = \emptyset$
2. *If $A \wedge B \in \{\!|X|\!\}$ then $A \in \{\!|X|\!\}$ and $B \in \{\!|X|\!\}$*
3. *If $A \wedge B \in \{\!|Y|\!\}$ then $A \in \{\!|Y|\!\}$ or $B \in \{\!|Y|\!\}$*
4. *If $A \vee B \in \{\!|X|\!\}$ then $A \in \{\!|X|\!\}$ or $B \in \{\!|X|\!\}$*
5. *If $A \vee B \in \{\!|Y|\!\}$ then $A \in \{\!|Y|\!\}$ and $B \in \{\!|Y|\!\}$*
6. *If $A \rightarrow B \in \{\!|X|\!\}$ then $A \in \{\!|Y|\!\}$ or $B \in \{\!|X|\!\}$*
7. *If $A \mathrel{-\!\!<} B \in \{\!|Y|\!\}$ then $A \in \{\!|Y|\!\}$ or $B \in \{\!|X|\!\}$*
8. *If $A \rightarrow B \in \{\!|Y|\!\}$ then $B \in \{\!|Y|\!\}$*
9. *If $A \mathrel{-\!\!<} B \in \{\!|X|\!\}$ then $A \in \{\!|X|\!\}$*

Let X and Y be two structures. We say that a formula $A \rightarrow B$ is *realised* by $X \triangleright Y$ iff either:

1. there exists a structure $Z \triangleright W \in Y$ such that $A \in \{\!|Z|\!\}$ and $B \in \{\!|W|\!\}$, or
2. $A \in \{\!|X|\!\}$ and $B \in \{\!|Y|\!\}$

We say that a formula $C \mathrel{-\!\!<} D$ is *realised* by $X \triangleright Y$ iff either:

1. there exists a structure $Z \triangleright W \in X$ such that $C \in \{\!|Z|\!\}$ and $D \in \{\!|W|\!\}$, or

2. $C \in \{|X|\}$ and $D \in \{|Y|\}$

We define the super-set relation on sequents as follows:

$$X_1 \triangleright Y_1 \supset X_0 \triangleright Y_0 \text{ iff } \{|X_1|\} \supset \{|X_0|\} \text{ or } \{|Y_1|\} \supset \{|Y_0|\}.$$

Then the following simple modifications of **DBiInt** ensure termination using only local checks:

Definition 5.3.3. *Let* $\mathbf{DBiInt_1}$ *be the system obtained from* **DBiInt** *with the following changes:*

1. *Add the derived rules* $-\!\!<_{L1}$ *and* $\rightarrow_{R1}$.

2. *Replace rules* $-\!\!<_L$, $\rightarrow_R$ *by the following:*

$$\frac{\Sigma^-[A -\!\!< B, (A \triangleright B)]}{\Sigma^-[A -\!\!< B]} -\!\!<_L$$

where $\Sigma^-[A -\!\!< B]$ *is saturated and* $A -\!\!< B$ *is not realised by* $\overbrace{\Sigma^-[A -\!\!< B]}$

$$\frac{\Sigma^+[A \rightarrow B, (A \triangleright B)]}{\Sigma^+[A \rightarrow B]} \rightarrow_R$$

where $\Sigma^+[A \rightarrow B]$ *is saturated and* $A \rightarrow B$ *is not realised by* $\overbrace{\Sigma^+[A \rightarrow B]}$

3. *Replace rules* $\triangleright_{L2}$ *and* $\triangleright_{R2}$ *by the following:*

$$\frac{\Sigma[A, X \triangleright (W, (A, Y \triangleright Z))]}{\Sigma[A, X \triangleright (W, (Y \triangleright Z))]} \triangleright_{L2} \textit{ where } A \notin \{|Y|\}$$

$$\frac{\Sigma[((X \triangleright Y, A), W) \triangleright Z, A]}{\Sigma[((X \triangleright Y), W) \triangleright Z, A]} \triangleright_{R2} \textit{ where } A \notin \{|Y|\}$$

4. *Replace rules* $\rightarrow_L$, $-\!\!<_R$, $-\!\!<_{L1}$, $\rightarrow_{R1}$, $\triangleright_{L1}$, $\triangleright_{R1}$, $\wedge_L$, $\wedge_R$, $\vee_L$, $\vee_R$ *with the following restricted versions:*

   (a) *Let* γ_0 *be the conclusion of the rule let* γ_1 *(and* γ_2*) be the premises. The rule is applicable only if:* $\overbrace{\gamma_1} \supset \overbrace{\gamma_0}$ *and* $\overbrace{\gamma_2} \supset \overbrace{\gamma_0}$.

We will now give a simple proof search strategy for $\mathbf{DBiInt_1}$. While traditional tableaux methods operate on a single node at a time, our proof search strategy will consider the whole tree. First we define a mapping from sequents to trees.

A *node* is a pair of sets of formulae. A *tree* is a node with 0 or more children, where each child is a tree, and each child is labelled as either a $\leq$-child, or a $\geq$-child. Given a sequent $\Xi = (X_1 \triangleright Y_1), \cdots, (X_n \triangleright Y_n), \Gamma \triangleright \Delta, (Z_1 \triangleright W_1), \cdots, (Z_m \triangleright W_m)$, where Γ and Δ are sets of formulae and $n \geq 0$ and $m \geq 0$, the tree *tree*(Ξ) represented by Ξ is:

```
Function Prove (Sequent $\Xi$) : Bool
  1. Let $T = tree(\Xi)$
  2. If the $id$, $\bot_L$, or $\top_R$ rule is applicable to any node in $T$, return $True$
  3. Else if there is some node $\langle \Gamma, \Delta \rangle \in T$ that is not saturated
     (a) Let $\rho$ be the rule corresponding to the requirement of Definition 5.3.2 that
         is not met, and let $\Xi_1$ (and $\Xi_2$) be the premise(s) of $\rho$. Return $\bigwedge Prove(\Xi_i)$.
  4. Else if there is some node $\Theta$ that is not propagated
     (a) Let $\rho$ be the rule corresponding to the requirement of Definition 5.3.4 that
         is not met, and let $\Xi_1$ be the premise of $\rho$. Return $Prove(\Xi_1)$.
  5. Else if there is some node $\langle \Gamma, \Delta \rangle \in T$ that is not realised, i.e. some $C = A \to B \in$
     $\Delta$ ($C = A \prec B \in \Gamma$) is not realised
     (a) Let $\Xi_1$ be the premise of the $\to_R$ ($\prec_L$) rule applied to $C \in \Delta$ ($C \in \Gamma$).
         Return $Prove(\Xi_1)$.
  6. Else return $False$
```

Figure 5.2: A proof search strategy for $\mathbf{DBiInt_1}$

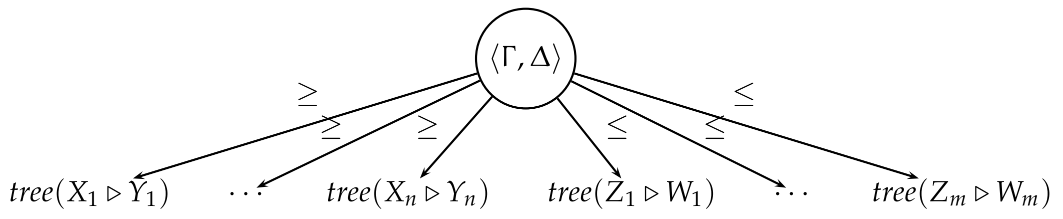

$\langle \Gamma, \Delta \rangle$
$\geq$
$\geq$
$\geq$
$\leq$
$\leq$
$\leq$
$tree(X_1 \triangleright Y_1)$
$\cdots$
$tree(X_n \triangleright Y_n)$
$tree(Z_1 \triangleright W_1)$
$\cdots$
$tree(Z_m \triangleright W_m)$

Figure 5.2 gives a proof search strategy for $\mathbf{DBiInt_1}$. The application of a rule deep inside a sequent can be viewed as focusing on a particular node of the tree. The rules of $\mathbf{DBiInt_1}$ can then be viewed as operations on the tree encoded in the sequent. In particular, Step 3 saturates a node locally, Step 4 propagates formulae between neighbouring nodes, and Step 5 appends new nodes to the tree.

Definition 5.3.4. *Given a tree T and a node $\Theta = \langle \Gamma, \Delta \rangle \in T$, we say $\langle \Gamma, \Delta \rangle$ is propagated iff:*

$\triangleright_{L1}$: *for every $\geq$-child $\Gamma_1 \triangleright \Delta_1$ of Θ and for every $A \in \Gamma_1$, we have $A \in \Gamma$*

$\triangleright_{R1}$: *for every $\leq$-child $\Gamma_1 \triangleright \Delta_1$ of Θ and for every $A \in \Delta_1$, we have $A \in \Delta$*

$\triangleright_{L2}$: *for every $A \in \Gamma$ and for every $\leq$-child $\Gamma_1 \triangleright \Delta_1$ of Θ, we have $A \in \Gamma_1$*

$\triangleright_{R2}$: *for every $A \in \Delta$ and for every $\geq$-child $\Gamma_1 \triangleright \Delta_1$ of Θ, we have $A \in \Delta_1$*

We will now show that *Prove* is complete for bi-intuitionistic logic. As is usual in completeness proofs, we will show the contrapositive: if *Prove* returns false for some sequent Ξ, then Ξ is not valid. Instead of a semantic completeness proof, we will use a purely syntactic method via the sound and complete calculus **DBiInt**. The following auxiliary lemma shows that if *Prove* returns false for some sequent Ξ, there is no **DBiInt** derivation of that sequent.

Lemma 5.3.5. *Let Ξ be a sequent such that every node in tree(Ξ) is saturated, realised and propagated. Then Ξ is not derivable in* **DBiInt**.

Proof. We prove the statement of the lemma by contradiction. That is, we assume every node in *tree*(Ξ) is saturated, realised and propagated, and that Ξ is derivable in **DBiInt**. Then there exists a shortest derivation Π of Ξ. We now consider all the rule instances that Π could end with, and in each case show that there is an even shorter **DBiInt** derivation of Ξ, therefore contradicting our assumption that Π was the shortest derivation.

- Π cannot end with the *id* rule, since every node in *tree*(Ξ) is saturated.
- Suppose Π ends with the $\vee_R$ rule, where $\Xi = \Sigma^+[A \vee B, A, B]$ for some $\Sigma^+[]$, and the last rule of Π applies to that particular occurrence of $A \vee B$ in the context $\Sigma^+[]$. Note that since every node of *tree*(Ξ) is saturated, we know that A and B are also present at the node where $A \vee B$ is located. Then Π must be of the form:

$$\dfrac{\begin{matrix}\Pi_1\\ \Sigma^+[A \vee B, A, B, A, B]\end{matrix}}{\Sigma^+[A \vee B, A, B]}\ \vee_R$$

  Now, applying Lemma 5.2.3 twice to Π_1, we obtain a derivation Π_2 of

$$\Sigma^+[A \vee B, A, B]$$

  such that $|\Pi_2| = |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\vee_R$.
- All other cases involving rules $\rightarrow_L$, $\prec_R$, $\wedge_L$, $\wedge_R$, $\vee_L$ can be treated analogously to $\vee_R$, using height preserving admissibility of formula contraction.
- Suppose Π ends with the $\rightarrow_R$ rule. There are two subcases, since every node in *tree*(Ξ) is realised: either $\Xi = \Sigma[X \triangleright A \rightarrow B, B, (W_1, A \triangleright B, Z_1)]$ for some $\Sigma[]$, or $\Xi = \Sigma[X_1, A \triangleright B, A \rightarrow B, Y_2]$ for some $\Sigma[]$.
  - Suppose $\Xi = \Sigma[X \triangleright A \rightarrow B, B, (W_1, A \triangleright B, Z_1)]$ for some $\Sigma[]$ and the last rule of Π applies to that particular occurrence of $A \rightarrow B$ in the context $\Sigma[]$. Then Π must be of the form:

$$\dfrac{\begin{matrix}\Pi_1\\ \Sigma[X \triangleright A \rightarrow B, B, (W_1, A \triangleright B, Z_1), (A \triangleright B)]\end{matrix}}{\Sigma[X \triangleright A \rightarrow B, B, (W_1, A \triangleright B, Z_1)]}\ \rightarrow_R$$

By the distribution Lemma 5.2.9, there is a **DBiInt** derivation Π_2 of

$$\Sigma[X \triangleright A \to B, B, (W_1, A, A \triangleright B, B, Z_1)]$$

such that $|\Pi_2| = |\Pi_1|$. Then applying Lemma 5.2.3 to Π_2 twice gives us a **DBiInt** derivation Π_3 of

$$\Sigma[X \triangleright A \to B, B, (W_1, A \triangleright B, Z_1)]$$

such that $|\Pi_3| = |\Pi_2| = |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\to_R$.

– Suppose $\Xi = \Sigma[X_1, A \triangleright B, A \to B, Y_2]$ for some $\Sigma[]$ and the last rule of Π applies to that particular occurrence of $A \to B$ in the context $\Sigma[]$. Then Π must be of the form:

$$\cfrac{\begin{matrix}\Pi_1 \\ \Sigma[X_1, A \triangleright B, A \to B, (A \triangleright B), Y_2]\end{matrix}}{\Sigma[X_1, A \triangleright B, A \to B, Y_2]} \to_R$$

By Lemma 5.2.6, we can obtain a derivation Π_2 of

$$\Sigma[X_1, A, A \triangleright B, B, A \to B, Y_2]$$

such that $|\Pi_2| \leq |\Pi_1|$. Then applying Lemma 5.2.3 to Π_2 twice gives us a **DBiInt** derivation Π_3 of

$$\Sigma[X_1, A \triangleright B, A \to B, Y_2]$$

such that $|\Pi_3| = |\Pi_2| \leq |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\to_R$.

- Suppose Π ends with the $-\!\!<_L$ rule. The proof is symmetric to the previous case, using Lemma 5.2.5 instead.
- Suppose Π ends with the $\triangleright_{L2}$ rule, where $\Xi = \Sigma[A, X \triangleright (W, (A, Y_1 \triangleright Z))]$ for some $\Sigma[]$, and the last rule of Π applies to that particular occurrence of A in the context $\Sigma[]$. Note that since every node of $tree(\Xi)$ is propagated, we know that A is also present at the inner node $A, Y_1 \triangleright Z$. Then Π must be of the form:

$$\cfrac{\begin{matrix}\Pi_1 \\ \Sigma[A, X \triangleright (W, (A, A, Y_1 \triangleright Z))]\end{matrix}}{\Sigma[A, X \triangleright (W, (A, Y_1 \triangleright Z))]} \triangleright_{L2}$$

Now, applying Lemma 5.2.3 to Π_1, we obtain a derivation Π_2 of

$$\Sigma[A, X \triangleright (W, (A, Y_1 \triangleright Z))]$$

such that $|\Pi_2| = |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\triangleright_{L2}$.

- All other cases involving propagation rules $\triangleright_{L1}$, $\triangleright_{R1}$, $\triangleright_{R2}$ can be treated analogously to $\triangleright_{L2}$, using height preserving admissibility of formula contraction.

Since Π cannot end with any of the rules of **DBiInt**, this obviously contradicts the assumption that it is a derivation in **DBiInt**. Therefore Ξ is not derivable in **DBiInt**. Q.E.D.

Theorem 5.3.6. *For any X and Y, $Prove(X \triangleright Y) = true$ if and only if $\vdash_{\mathbf{DBiInt}} \Pi : X \triangleright Y$.*

Proof.

- Left-to-right: obvious, since every step of *Prove* is a backwards application of a $\mathbf{DBiInt_1}$ rule, which is either a (possibly restricted) rule of **DBiInt** or a derived rule.

- Right-to-left: we show that if $Prove(X \triangleright Y)$ returns False then $X \triangleright Y$ is not derivable in **DBiInt**. Since each rule of **DBiInt** is invertible (Lemma 5.2.4), Steps 2 to 5 of *Prove* preserve provability of the original sequent. If $Prove(X \triangleright Y)$ returns False, this can only be the case if Step 6 is reached, i.e., the systematic bottom-up applications of the rules of $\mathbf{DBiInt_1}$ produce a sequent such that every node in the tree of the sequent is saturated, realised, and propagated. By Lemma 5.3.5, such a sequent would not be derivable in **DBiInt**, and since all other steps of *Prove* preserve derivability, it follows that $X \triangleright Y$ is not derivable either in **DBiInt**.

Q.E.D.

Before proving termination formally, we illustrate how the non-terminating backward derivation attempt of Example 5.3.1 is now blocked by $\mathbf{DBiInt_1}$.

Example 5.3.7. *For readability, we abbreviate $X = (A \to B) \to C, (D \to E) \to F$, $Y = A \to B, D \to E, B, E$ and $\Sigma[] = X \triangleright G, Y, []$.*

$$
\begin{array}{c}
\textit{no rule applicable} \\
\Sigma[(X, A \rhd Y, \boxed{(X, A, D \rhd E, A \to B, D \to E, B)}), (X, D \rhd Y, \boxed{(X, D, A \rhd B, A \to B, D \to E, E)})] \\
\hline
\Sigma[(X, A \rhd Y, \boxed{(X, A, D \rhd E, A \to B, D \to E)}), (X, D \rhd Y, \boxed{(X, D, A \rhd B, A \to B, D \to E)})] \\
\hline
\Sigma[\boxed{(X, A \rhd Y, (X, A, D \rhd E))}, \boxed{(X, D \rhd Y, (X, D, A \rhd B))}] \\
\hline
\Sigma[\boxed{(X, A \rhd Y, (D \rhd E))}, \boxed{(X, D \rhd Y, (A \rhd B))}] \\
\hline
\Sigma[\boxed{(X, A \rhd B, A \to B, D \to E, E)}, \boxed{(X, D \rhd E, A \to B, D \to E, B)}] \\
\hline
\Sigma[\boxed{(X, A \rhd B, A \to B, D \to E)}, \boxed{(X, D \rhd E, A \to B, D \to E)}] \\
\hline
X \rhd G, Y, \boxed{(X, A \rhd B)}, \boxed{(X, D \rhd E)} \\
\hline
(A \to B) \to C, (D \to E) \to F \rhd G, A \to B, D \to E, B, E, (A \rhd B), (D \rhd E) \\
\hline
(A \to B) \to C, (D \to E) \to F \rhd G, A \to B, D \to E, B, E \\
\hline
(A \to B) \to C, (D \to E) \to F \rhd G, A \to B, D \to E \\
\hline
(A \to B) \to C, (D \to E) \to F \rhd G
\end{array}
\begin{array}{l}
\\ \\
2\times \to_{R1} \\ \\
2\times \to_{L} \\ \\
6 \times \rhd_{L2} \\ \\
2\times \to_{R} \\ \\
2\times \to_{R1} \\ \\
4\times \to_{L} \\ \\
4 \times \rhd_{L2} \\
2\times \to_{R} \\
2\times \to_{R1} \\
2\times \to_{L}
\end{array}
$$

Note that no rules are applicable backwards to the sequent

$$\Sigma[(X, A \rhd Y, \boxed{(X, A, D \rhd E, A \to B, D \to E, B)}), (X, D \rhd Y, \boxed{(X, D, A \rhd B, A \to B, D \to E, E)})]$$

at the top of the derivation attempt. In particular, the previously used $\to_R$ is not applicable to $A \to B$ in either of the contexts illustrated by the boxes, since A is on the left hand side and B is on the right hand side and so the formula $A \to B$ is realised by the context. Similarly, $\to_R$ is not applicable to $D \to E$ in either of the contexts illustrated by the boxes.

We write $sf(A)$ for the subformulae of A, and define the set of subformulae of a set Θ as $sf(\Theta) = \bigcup_{A \in \Theta} sf(A)$. For a sequent Ξ we define $sf(\Xi)$ as below:

$$
\begin{array}{rcl}
\Xi & = & (X_1 \rhd Y_1), \cdots, (X_n \rhd Y_n), \Gamma \rhd \Delta, (Z_1 \rhd W_1), \cdots, (Z_m \rhd W_m) \\
sf(\Xi) & = & sf(\Gamma) \cup sf(\Delta) \\
 & \cup & sf(X_1 \rhd Y_1) \cup \cdots \cup sf(X_n \rhd Y_n) \\
 & \cup & sf(Z_1 \rhd W_1) \cup \cdots \cup sf(Z_m \rhd W_m).
\end{array}
$$

The idea behind the termination proof is essentially the same as the one we used for $\mathbf{LBiInt_2}$ in Chapter 4. Here we show how to apply our method to the deep inference nested sequent calculus $\mathbf{DBiInt_1}$ instead. The key idea again is to show that *Prove* cannot create trees of infinite depth, in particular, that it cannot create infinitely many zig-zags of $\leq$ and $\geq$ edges. To see why this is the case intuitively, consider the propagation rules $\rhd_{L2}$ and $\rhd_{R2}$, which are the only rules that move formulae away from the root of the tree towards the leaves. Note that $\rhd_{L2}$ can only move formulae along $\leq$ edges and place them on the left hand side of child nodes. Similarly, $\rhd_{R2}$ can only move formulae along $\geq$ edges and place them on the right hand side of child nodes. Therefore after a zig-zag of $\leq$ and $\geq$ edges some formulae are "stuck" and cannot move any further: see Figure 5.3.

$$\begin{tikzcd}
\langle \{A, \Gamma_0\}, \Delta_0 \rangle \arrow[d, "\leq"] \arrow[d, dotted, bend right=60, "A"'] \\
\langle \Gamma_1, \{\Delta_1, B\} \rangle \arrow[d, "\geq"] \arrow[d, dotted, bend left=60, "B"] \\
\langle \Gamma_2, \Delta_2 \rangle
\end{tikzcd}$$

Figure 5.3: Propagation of formulae. A is propagated along the $\leq$ edge between the left hand sides of nodes, and B is propagated along the $\geq$ edge between the right hand sides of nodes. Note that A cannot be propagated to the bottom-most node.

We now define the degree of a node in a nested sequent, that is, the degree of a particular structure placed in a certain context. The following definition is essentially the same as Definition 4.4.8, but here we apply it to some specific node within a nested sequent, rather than the top level formulae only.

Definition 5.3.8. *The degree of a formula is:*

$$\begin{array}{rcl}
degf(p) & = & 0 \\
degf(A \land B) = degf(A \lor B) & = & max(degf(A), degf(B)) \\
degf(A \to B) = degf(A \prec B) & = & 1 + max(degf(A), degf(B)).
\end{array}$$

Given a sequent $\Sigma[Z]$*, we define the degree of* $\Sigma[Z]$ *w.r.t.* $\Sigma[]$*. Let* $X \rhd Y = \widehat{\Sigma[Z]}$*. Then:*

$$\begin{array}{rcl}
deg_L(X \rhd Y) & = & max\{degf(A) \mid A \in \{|X|\}\} \\
deg_R(X \rhd Y) & = & max\{degf(B) \mid B \in \{|Y|\}\} \\
deg(X \rhd Y) & = & max(deg_L(X \rhd Y), deg_R(X \rhd Y)) \\
degc(\Sigma[Z]_{\Sigma[]}) & = & deg(\widehat{\Sigma[Z]}) = deg(X \rhd Y).
\end{array}$$

Note also that our definition of a degree of a node nested within a tree bears similarities to Brünnler's work on deep inference calculi for modal logics [20]. There, he obtains termination by using a global loop check, which examines nodes nested in the tree of nested sequents. His proof search procedure blocks the application of rules to "cyclic" nodes in the tree, where a leaf of a sequent is defined as cyclic if there is an inner node in the sequent that carries the same set of formulas [20].

Theorem 5.3.9. *For any two sets of* `BiInt`*-formulae* Γ *and* Δ*, Prove*$(\Gamma \rhd \Delta)$ *terminates.*

Proof. Let $m = |sf(\Gamma \rhd \Delta)|$. To show that *Prove* terminates, we will argue about the

$$\dfrac{\begin{array}{c}\Sigma^+[A \to B, B, (X', A \rhd B, A \to B, Y')] \\ \vdots \\ \Sigma^+[A \to B, B, (A \rhd B)]\end{array}}{\Sigma^+[A \to B, B]} \to_R$$

Figure 5.4: Blocked repeated application of $\to_R$. Note that an $\to_R$ application to $A \to B$ is blocked for the topmost sequent because $A \to B$ already realised by $(X', A \rhd B, A \to B, Y')$. The dotted part of the derivation may include any propagation rules, static rules or other $\to_R$ rule instances.

tree $T = tree(X \rhd Y)$, where $X \rhd Y$ is the parameter to the most recent recursive call to *Prove*. That is, initially $T = tree(\Gamma \rhd \Delta)$.

The saturation process for each node in T is bounded by m. Therefore after at most m moves at each node, Step 3 is no longer applicable to this node because of restrictions 3 and 4 of Definition 5.3.3. Since formulae are only propagated to nodes that do not already contain these formulae, after at most m propagation moves into each node, Step 4 is no longer applicable to this node.

T is finitely branching, since new nodes are only created for unrealised $\to$- and $\prec$-formulae. Therefore after at most m moves at each node, Step 5 is no longer applicable to this node. We will now show that the depth of T is bounded by m^2, using a very similar argument to that of Lemma 4.4.9, but taking into account nested sequents and our new definition of degree.

First, we show that *Prove* can create at most m consecutive nodes in the same direction, i.e., where the links are all labeled by $\leq$ or all labeled by $\geq$. In the $\leq$ case, we need to show that we cannot create a node twice for the same $\to$-formula. Consider an application of $\to_R$ with principal formula $A \to B$. After this application, the structure $(A \rhd B)$ will be added to the nested sequent and saturated as well as propagated. During this process, $A \to B$ may reappear on the right hand side of the new structure: see Figure 5.4. However, then B will also be added to the right hand side of the structure by the $\to_{R1}$ rule during saturation, so a repeated application of $\to_R$ to $A \to B$ will be blocked by the side condition of $\to_R$, since $A \to B$ is now realised by the structure. Thus since the number of $\to$-formulae is bounded by m and we can only create a new node for each $\to$-formula once, there can be at most m consecutive nodes where the links are all labeled by $\leq$. The case for $\geq$ is symmetric.

We now show that we can switch direction at most m times. Consider a direction switch, e.g., a new node created using $\to_R$ followed by a new node created using $\prec_{L2}$ (the other case is symmetric), and any saturation and propagation rule applications in between. The following derivation fragment illustrates such a direction switch scenario, where Σ is any context and the vertical dots indicate any number of saturation and propagation steps:

$$\dfrac{\dfrac{\dfrac{\gamma_3 = \Sigma[X'_0 \triangleright Y'_0, A \to B, (X'_1, C \prec D, (X_2, C \triangleright D, Y_2), A \triangleright B, Y'_1)]}{\vdots}}{\dfrac{\Sigma[X'_0 \triangleright Y'_0, A \to B, (X_1, C \prec D, (C \triangleright D), A \triangleright B, Y_1)]}{\gamma_2 = \Sigma[X'_0 \triangleright Y'_0, A \to B, (X_1, C \prec D, A \triangleright B, Y_1)]} \prec_L}{\vdots}}{\dfrac{\gamma_1 = \Sigma[X_0 \triangleright Y_0, A \to B, (A \triangleright B)]}{\gamma_0 = \Sigma[X_0 \triangleright Y_0, A \to B]} \to_R}$$

We will now compare the degrees of sequent γ_3 with respect to various contexts, each going deeper inside the nested structure, which corresponds to being further away from the root of the tree. We want to show that the degree of the frontier nodes (i.e. furthest from the root) is smaller than the degree of nodes closer to the root of tree.

We first define the degree of the node closest to the root of the tree:

$$\begin{aligned} d_0 &= degc(\Sigma[X'_0 \triangleright Y'_0, A \to B, (X'_1, C \prec D, (X_2, C \triangleright D, Y_2), A \triangleright B, Y'_1)]_{\Sigma[]}) \\ &= deg(X'_0 \triangleright Y'_0, A \to B) \end{aligned}$$

Now we define two additional contexts, corresponding to the nodes being created by the $\to_R$ and $\prec_L$ rule applications. That is $\Sigma_1 = \Sigma[X_0 \triangleright Y_0, A \to B, []]$ and $\Sigma_2 = \Sigma[X'_0 \triangleright Y'_0, A \to B, (X_1, C \prec D, [], A \triangleright B, Y_1)]$. This lets us define the degrees of the child nodes:

$$\begin{aligned} d_1 &= degc(\Sigma_1[X'_1, C \prec D, A \triangleright B, Y'_1]_{\Sigma_1}) \\ &= deg(X'_1, C \prec D, A \triangleright B, Y'_1) \end{aligned}$$

and

$$\begin{aligned} d_2 &= degc(\Sigma_2[X_2, C \triangleright D, Y_2]_{\Sigma_2}) \\ & \quad deg(X_2, C \triangleright D, Y_2) \end{aligned}$$

We will now show that $d_2 \leq d_0 - 1$.

First, we have that $deg_L(X'_1, C \prec D, A \triangleright B, Y'_1) \leq deg_L(X'_0 \triangleright Y'_0, A \to B)$ because $\triangleright_{L2}$ rule applications can propagate all formulae from X'_0 into X'_1. However, since there are no propagation rules that could move formulae from outer contexts into the right hand side of $X_1', C \prec D, A \triangleright B, Y_1'$, we have that

$$\begin{array}{ll} & deg_R(X_1', C \prec D, A \triangleright B, Y_1') \\ \leq & max\{deg_L(X_1', C \prec D, A \triangleright B, Y_1') - 1, deg_R(X_0' \triangleright Y_0', A \to B) - 1\} \\ \leq & deg(X_0' \triangleright Y_0', A \to B) - 1 \end{array}$$

Secondly, we have that $deg_R(X_2, C \triangleright D, Y_2) \leq deg_R(X_1', C \prec D, A \triangleright B, Y_1')$ because $\triangleright_{R2}$ rule applications can propagate all formulae from Y_1' into Y_2. That is, $deg_R(X_2, C \triangleright D, Y_2) \leq deg(X_0' \triangleright Y_0', A \to B) - 1$. However, since there are no propagation rules that could move formulae from outer contexts into the left hand side of $X_2, C \triangleright D, Y_2$, we have that

$$\begin{array}{ll} & deg_L(X_2, C \triangleright D, Y_2) \\ \leq & max\{deg_L(X_1', C \prec D, A \triangleright B, Y_1') - 1, deg_R(X_2, C \triangleright D, Y_2) - 1\} \\ \leq & deg(X_0' \triangleright Y_0', A \to B) - 1 \end{array}$$

Summing up, we have

$$\begin{array}{rcl} deg(X_2, C \triangleright D, Y_2) & = & max\{deg_L(X_2, C \triangleright D, Y_2), deg_R(X_2, C \triangleright D, Y_2\} \\ & \leq & max\{deg(X_0' \triangleright Y_0', A \to B) - 1, deg(X_0' \triangleright Y_0', A \to B) - 1\} \\ & \leq & deg(X_0' \triangleright Y_0', A \to B) - 1 \end{array}$$

That is, $d_2 \leq d_0 - 1$.

After a direction switch, we can again make at most m node creation steps in one direction. Therefore the total number of node creation rule applications is bounded by $\mathcal{O}(m^2)$. Q.E.D.

Chapter 6

Shallow and deep inference nested calculi for tense logic

In this chapter we apply the methodology of the previous two chapters to give nested sequent calculi for tense logic and some its extensions. We use Kashima's calculi for tense logics [73] as a starting point for our proof theoretic (as opposed to the model-theoretic approach of Kashima) investigation of tense logic. Specifically, we show how deep inference in a nested sequent calculus can mimic residuation in Kashima's shallow inference calculus.

We begin in Section 6.1 with Kashima's first calculus **SKt** which contains structural connectives (proxies) for $\Diamond$ and $\blacklozenge$ and contains explicit "turn" rules to capture the residuation conditions that hold between them. Kashima shows that **SKt** is sound with respect to the Kripke semantics for tense logic, but he does not prove cut-admissibility for this calculus. He instead gives another calculus **S2Kt** which allows rules to be applied at arbitrary depth, and shows that a sequent has a cut-free derivation in **SKt** if it has a cut-free derivation in **S2Kt**. In a second step, he shows that **S2Kt** minus cut is complete w.r.t. the Kripke semantics of tense logic, which together imply the completeness of **SKt** minus cut.

We first replace formula contraction with general contraction in Kashima's **SKt**, show that the resulting calculus enjoys a display property, and show that it also has cut-admissibility using an argument which is very similar to Belnap's cut-admissibility proof for display calculi. We then show in Section 6.2 that Kashima's **S2Kt** minus cut (in the form of our **DKt**) can be made contraction-free and that the residuation rules of **SKt** are admissible in **DKt**, meaning that **DKt** can faithfully mimic cut-free **SKt**. We also show that **SKt** can mimic **DKt** by showing that all of the rules of **DKt** are actually derivable in **SKt** using the display property of **SKt**. In Section 6.3, we then show how to extend all these basic calculi to handle tense S4 and S5. Finally, we give a simple proof search strategy for **DKt** in Section 6.4.

Note. Some of the results of this chapter have been published in [58].

$$\frac{}{X, a, \bar{a}}\ id \qquad \frac{X, A \quad Y, \overline{A}}{X, Y}\ cut \qquad \frac{X, A \quad X, B}{X, A \wedge B}\ \wedge \qquad \frac{X, A, B}{X, A \vee B}\ \vee$$

$$\frac{X, Y, Y}{X, Y}\ ctr \qquad \frac{X}{X, Y}\ wk \qquad \frac{X, \circ\{Y\}}{\bullet\{X\}, Y}\ rf \qquad \frac{X, \bullet\{Y\}}{\circ\{X\}, Y}\ rp$$

$$\frac{X, \bullet\{A\}}{X, \blacksquare A}\ \blacksquare \qquad \frac{X, \circ\{A\}}{X, \Box A}\ \Box \qquad \frac{X, \bullet\{Y, A\}}{X, \bullet\{Y\}, \blacklozenge A}\ \blacklozenge \qquad \frac{X, \circ\{Y, A\}}{X, \circ\{Y\}, \Diamond A}\ \Diamond$$

Figure 6.1: **SKt**: a shallow inference nested sequent calculus for Kt

6.1 SKt: a shallow inference nested sequent calculus

To simplify presentation, we shall consider formulae of tense logic Kt which are in negation normal form (nnf), given by the following grammar:

$$A := a \mid \neg a \mid A \vee A \mid A \wedge A \mid \Box A \mid \blacksquare A \mid \Diamond A \mid \blacklozenge A.$$

where a ranges over atomic formulae and $\neg a$ is the negation of a. We shall denote with $\overline{A}$ the nnf of the negation of A. Implication can then be defined via negation: $A \rightarrow B = \overline{A} \vee B$.

Because we are dealing with classical tense logic, in this chapter we consider a right-sided sequent calculus for tense logic where the syntactic judgment is a tree of multisets of formulae, as has been done previously in proof systems for modal and tense logics [73; 17; 20].

Definition 6.1.1. *A* nested sequent *is a multiset*

$$\{A_1, \ldots, A_k, \circ\{X_1\}, \ldots, \circ\{X_m\}, \bullet\{Y_1\}, ..., \bullet\{Y_n\}\}$$

where $k, m, n \geq 0$, and each X_i and each Y_j are themselves nested sequents.

We shall use the following notational conventions when writing nested sequents. We shall remove outermost braces, e.g., we write A, B, C instead of $\{A, B, C\}$. Braces for sequents nested inside $\circ\{\}$ or $\bullet\{\}$ are also removed, for example, instead of writing $\circ\{\{A, B, C\}\}$, we write $\circ\{A, B, C\}$. When we juxtapose two sequents, e.g., as in X, Y, we mean it is a sequent resulting from the multiset-union of X and Y. When Y is a singleton multiset, e.g., $\{A\}$ or $\{\circ\{Y'\}\}$, we simply write: X, A or $X, \circ\{Y'\}$. Since we shall only be concerned with nested sequents, we shall refer to nested sequents simply as sequents in the rest of the chapter.

The above definition of sequents can also be seen as a special case of *structures* in display calculi, e.g., with ',' (comma), $\bullet$ and $\circ$ as structural connectives.

Similarly to previous chapters, a *context* is a sequent with holes in place of formulae. A context with a single hole is written as $\Sigma[]$. Multiple-hole contexts are written as $\Sigma[] \cdots []$, or abbreviated as $\Sigma^k[]$ where k is the number of holes. We write $\Sigma^k[Y]$ to denote the sequent that results from filling the holes in $\Sigma^k[]$ uniformly with Y.

The shallow sequent calculus for Kt, called **SKt**, is given in Figure 6.1. This is basically Kashima's calculus (also called **SKt**) [73], but with a more general contraction rule (*ctr*), which allows contraction of arbitrary sequents. The modal fragment of **SKt** was also developed independently by Brünnler [17; 20]. The general contraction rule is used to simplify our cut elimination proof, and as we shall see in Section 6.2, it can be replaced by formula contraction. The calculus **SKt** can also be seen as a single-sided version of display calculus. The rules rp and rf are called the *residuation rules*. They are an example of *display postulates* commonly found in display calculus, and are used to bring a node in a nested sequent to the top level. The following lemma and proposition show the analog of the display property of display calculus.

Lemma 6.1.2. *Let $\Sigma_1[X], \Sigma_2[Y]$ be a sequent with a unique occurrence of a structure X and a unique occurrence of a structure Y (X and Y are distinct structures). Then there exists a context $\Sigma[]$ such that $X, \Sigma[Y]$ is derivable from $\Sigma_1[X], \Sigma_2[Y]$ using only the rules rp and rf.*

Proof. By induction on the size of the context $\Sigma_1[]$. The non-trivial cases are when $\Sigma_1[]$ is of the form $\circ\{\Sigma_3[]\}$ or $\bullet\{\Sigma_3[]\}$. We consider the former case here; the latter can be handled analogously. By induction hypothesis, there exists $\Sigma'[]$ such that $X, \Sigma'[Y]$ is derivable from $\Sigma_3[X], \bullet\{\Sigma_2[Y]\}$. Let $\Sigma[] = \Sigma'[]$. Then we have a derivation

$$\cfrac{\circ\{\Sigma_3[X]\}, \Sigma_2[Y]}{\cfrac{\Sigma_3[X], \bullet\{\Sigma_2[Y]\}}{\begin{array}{c}\vdots\\ X, \Sigma'[Y]\end{array}}}\ rf$$

that uses only rp and rf. Q.E.D.

Proposition 6.1.3. *Let $\Sigma[W]$ be a sequent. Then there exists a sequent Z such that W, Z is derivable from $\Sigma[W]$ and vice versa, using only the rules rp and rf.*

Proof. We first show the forward direction, i.e., deriving W, Z from $\Sigma[W]$. Applying Lemma 6.1.2 with $\Sigma_1[] = \Sigma[]$ and $\Sigma_2[] = []$ and $X = W$ and $Y = \{\}$, we have a context $\Sigma'[]$ such that there exists a derivation Π from $\Sigma[W], \{\}$ to $W, \Sigma'[\{\}]$, using only rp and rf. That is:

$$\begin{array}{c}\Sigma[W], \{\}\\ \vdots\\ W, \Sigma'[\{\}]\end{array}$$

Let $Z = \Sigma'[\{\}]$. Now we get a derivation of W, Z from $\Sigma[W]$ using only rp and rf. That is:

$$\begin{array}{c}\Sigma[W]\\ \vdots\\ W, Z\end{array}$$

For the converse, we first observe that the rules rp and rf are dual to each other. That is, if we turn the rule rp upside down, we get rf. Therefore, to get a derivation of $\Sigma[W]$ from W, Z, we just turn the derivation obtained in the previous case upside down, and rename the rule rp to rf and vice versa. Q.E.D.

6.1.1 Soundness and completeness

To prove soundness, we first show that each sequent has a corresponding Kt-formula, and then show that the rules of **SKt**, reading them top down, preserve validity of the formula corresponding to the premise sequent. Completeness is shown by simulating Hilbert's system for tense logic in **SKt**. The translation from sequents to formulae is given below. In the translation, we assume two logical constants $\bot$ ('false') and $\top$ ('true'). This is just a notational convenience, as the constants can be defined in a standard way, e.g., as $a \wedge \bar{a}$ and $a \vee \bar{a}$ for some fixed atomic proposition a.

Definition 6.1.4. *The function τ translates an* **SKt***-sequent*

$$\{A_1, \dots, A_k, \circ\{X_1\}, \dots, \circ\{X_m\}, \bullet\{Y_1\}, ..., \bullet\{Y_n\}\}$$

into the Kt*-formula (modulo associativity and commutativity of $\vee$ and $\wedge$):*

$$A_1 \vee \cdots \vee A_k \vee \Box\tau(X_1) \vee \cdots \vee \Box\tau(X_m) \vee \blacksquare\tau(Y_1) \vee \cdots \vee \blacksquare\tau(Y_n).$$

As usual, the empty disjunction denotes $\bot$.

Theorem 6.1.7 is a simple corollary of the following lemmas.

Lemma 6.1.5 (Soundness)**.** *Every* **SKt***-derivable* Kt *formula is valid.*

Proof. We show that for every rule ρ of **SKt**

$$\frac{X_1 \quad \cdots \quad X_n}{X}\ \rho$$

the following holds: if for every $i \in \{1, \dots, n\}$, the formula $\tau(X_i)$ is valid then the formula $\tau(X)$ is valid. The following are the formula translations for the residuation rules and tense/modal rules:

rf**:** if $\tau(X) \vee \Box(\tau(Y))$ valid then $\blacksquare(\tau(X)) \vee \tau(Y)$ valid

rp**:** if $\tau(X) \vee \blacksquare(\tau(Y))$ valid then $\Box(\tau(X)) \vee \tau(Y)$ valid

$\blacksquare$**:** if $\tau(X) \vee \blacksquare A$ valid then $\tau(X) \vee \blacksquare A$ valid

$\Box$**:** if $\tau(X) \vee \Box A$ valid then $\tau(X) \vee \Box A$ valid

$\blacklozenge$**:** if $\tau(X) \vee \blacksquare(\tau(Y) \vee A)$ valid then $\tau(X) \vee \blacksquare(\tau(Y)) \vee \blacklozenge A$ valid

$\Diamond$**:** if $\tau(X) \vee \Box(\tau(Y) \vee A)$ valid then $\tau(X) \vee \Box(\tau(Y)) \vee \Diamond A$ valid

We now prove the cases for rf and $\blacklozenge$ in detail, the others are analogous or easier.

- Case rf: we assume that $\tau(X) \vee \Box(\tau(Y))$ is valid, that is:

  $$\forall \langle W, R, V \rangle \forall w \in W \,.\, w \Vdash \tau(X) \text{ or } w \Vdash \Box(\tau(Y)) \tag{6.1.1}$$

  More specifically, it means

  $$\forall \langle W, R, V \rangle \forall w \in W \,.\, w \Vdash \tau(X) \text{ or } \forall u \in W \text{ if } wRu \text{ then } u \Vdash \tau(Y) \tag{6.1.2}$$

  Now we need to show that $\blacksquare(\tau(X)) \vee \tau(Y)$ is valid. We do so by contradiction; that is, we assume that $\blacksquare(\tau(X)) \vee \tau(Y)$ is falsifiable:

  $$\exists \langle W', R', V' \rangle \exists u' \in W' \,.\, u' \nVdash \blacksquare(\tau(X)) \text{ and } u' \nVdash \tau(Y) \tag{6.1.3}$$

  More specifically, 6.1.3 also means that

  $$\exists w' \in W' \,. w'Ru' \text{ and } w' \nVdash \tau(X) \tag{6.1.4}$$

  However, now we have contradiction, since we have $w' \nVdash \tau(X) \vee \Box(\tau(Y))$, which we assumed to be valid. Therefore it cannot be the case that $\blacksquare(\tau(X)) \vee \tau(Y)$ is falsifiable, indeed $\blacksquare(\tau(X)) \vee \tau(Y)$ is valid.

- Case $\blacklozenge$: we assume that $\tau(X) \vee \blacksquare(\tau(Y) \vee A)$ is valid, that is:

  $$\forall \langle W, R, V \rangle \forall w \in W \,.\, w \Vdash \tau(X) \text{ or } w \Vdash \blacksquare(\tau(Y) \vee A) \tag{6.1.5}$$

  More specifically, it means

  $$\forall \langle W, R, V \rangle \forall w \in W \,.\, w \Vdash \tau(X) \text{ or } \forall u \in W \text{ if } uRw \text{ then } u \Vdash \tau(Y) \vee A \tag{6.1.6}$$

  Now we need to show that $\tau(X) \vee \blacksquare(\tau(Y)) \vee \blacklozenge A$ is valid. We do so by contradiction; that is, we assume that $\tau(X) \vee \blacksquare(\tau(Y)) \vee \blacklozenge A$ is falsifiable:

  $$\exists \langle W', R', V' \rangle \exists w' \in W' \,.\, w' \nVdash \tau(X) \text{ and } w' \nVdash \blacksquare(\tau(Y)) \text{ and } w' \nVdash \blacklozenge A \tag{6.1.7}$$

  More specifically, it means $\exists u' \in W'$ such that $u'Rw'$ and:

  $$u' \nVdash \tau(Y) \tag{6.1.8}$$
  $$u' \nVdash A \tag{6.1.9}$$

  However, the latter two contradict 6.1.6. Therefore it cannot be the case that $\tau(X) \vee \blacksquare(\tau(Y)) \vee \blacklozenge A$ is falsifiable, indeed $\tau(X) \vee \blacksquare(\tau(Y)) \vee \blacklozenge A$ is valid.

Since the formula-translation $\tau(X) \vee a \vee \bar{a}$ of the *id* rule is obviously valid, it then follows that every formula derivable in **SKt** is also valid. Q.E.D.

Lemma 6.1.6 (Completeness)**.** *Every* Kt*-theorem is* **SKt***-derivable.*

Proof. The following are derivations of the negation normal forms of Axioms 2.2.30 and 2.2.32, the other axioms are analogous. Dashed lines abbreviate derivations:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A\wedge\overline{B},A,\overline{A},B,\bullet\{\ \}}\,id\qquad\dfrac{}{A\wedge\overline{B},\overline{B},\overline{A},B,\bullet\{\ \}}\,id}{A\wedge\overline{B},\overline{A},B,\bullet\{\ \}}\,\wedge}{\circ\{A\wedge\overline{B},\overline{A},B\}}\,rp}{\Diamond(A\wedge\overline{B}),\Diamond\overline{A},\circ\{B\}}\,\Diamond}{\Diamond(A\wedge\overline{B}),\Diamond\overline{A},\Box B}\,\Box}{\Diamond(A\wedge\overline{B})\vee\Diamond\overline{A}\vee\Box B}\,\vee
\qquad
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\circ\{\ \},\overline{A},A}\,id}{\bullet\{\overline{A},A\}}\,rf}{\bullet\{\overline{A}\},\blacklozenge A}\,\blacklozenge}{\overline{A},\circ\{\blacklozenge A\}}\,rp}{\overline{A},\Box\blacklozenge A}\,\Box}{\overline{A}\vee\Box\blacklozenge A}\,\vee$$

The following are derivations of the negation normal forms of rules MP, $Nec\Box$ and $Nec\blacksquare$:

$$\dfrac{\overline{A}\vee B\qquad\dfrac{\dfrac{A}{A,B}\,wk\qquad\dfrac{}{\overline{B},B}\,id}{A\wedge\overline{B},B}\,\wedge}{B}\,cut$$

$$\dfrac{\dfrac{\dfrac{A}{A,\bullet\{\ \}}\,wk}{\circ\{A\}}\,rp}{\Box A}\,\Box\qquad\dfrac{\dfrac{\dfrac{A}{A,\circ\{\ \}}\,wk}{\bullet\{A\}}\,rf}{\blacksquare A}\,\blacksquare$$

Q.E.D.

Theorem 6.1.7. *A* Kt*-formula A is valid iff A is* **SKt***-derivable.*

Proof. By Lemmas 6.1.5 and 6.1.6. Q.E.D.

6.1.2 Cut elimination

The main difficulty in proving cut elimination[1] for **SKt** is in finding the right cut reduction for some cases involving the rules rp and rf. For instance, consider the derivation (1) in Figure 6.2. It is not obvious that there is a cut reduction strategy that works locally without generalizing the cut rule to, e.g., one which allows cut on any sub-sequent in a sequent. Instead, we shall follow a global cut reduction strategy similar to that used in cut elimination for display logics. The idea is that, instead of permuting the cut rule locally, we trace the cut formula A (in Π_1) and $\overline{A}$ (in Π_2), until they both become principal in their respective derivations, and then apply the cut rule(s) at that point on smaller formulae. Schematically, our simple strategy can be illustrated as follows: Suppose that Π_1 and Π_2 are, respectively, derivation (2) and (3) in Figure 6.2, that $A = A_1 \wedge A_2$ and there is a single instance in each derivation where the cut formula is used. To reduce the cut on A, we first transform Π_1 by uniformly substituting $\bullet\{Y\}$ for A in Π_1 (see derivation (4) in Figure 6.2). We then prove the open leaf $\{\circ\{\circ\{X'\}\},Y\}$ by uniformly substituting $\circ\{X'\}$ for A in Π_2 (see derivation

[1]The cut-elimination proof in this chapter is due to Alwen Tiu, and is included in this thesis for completeness.

$$\dfrac{\dfrac{\begin{matrix}\Pi_1\\ X, \bullet\{A\}\end{matrix}}{\circ\{X\}, A}\,rf \qquad \dfrac{\begin{matrix}\Pi_2\\ \circ\{\overline{A}\}, Y\end{matrix}}{\overline{A}, \bullet\{Y\}}\,rp}{\circ\{X\}, \bullet\{Y\}}\,cut$$

(1)

$$\begin{array}{c}\dfrac{\dfrac{\begin{matrix}\vdots\\ \circ\{X'\}, A_1\end{matrix}\quad\begin{matrix}\vdots\\ \circ\{X'\}, A_2\end{matrix}}{\circ\{X'\}, A_1 \wedge A_2}\,\wedge}{X', \bullet\{A_1 \wedge A_2\}}\,rf\\ \vdots\\ X, \bullet\{A_1 \wedge A_2\}\end{array}$$

(2)

$$\begin{array}{c}\vdots\\ \dfrac{\dfrac{\overline{A}_1, \overline{A}_2, \bullet\{Y'\}}{\overline{A}_1 \vee \overline{A}_2, \bullet\{Y'\}}\,\vee}{\circ\{\overline{A}_1 \vee \overline{A}_2\}, Y'}\,rp\\ \vdots\\ \circ\{\overline{A}_1 \vee \overline{A}_2\}, Y\end{array}$$

(3)

$$\begin{array}{c}\dfrac{\dfrac{\circ\{\circ\{X'\}\}, Y}{\circ\{X'\}, \bullet\{Y\}}\,rf}{X', \bullet\{\bullet\{Y\}\}}\,rf\\ \vdots\\ \dfrac{X, \bullet\{\bullet\{Y\}\}}{\circ\{X\}, \bullet\{Y\}}\,rp\end{array}$$

(4)

$$\begin{array}{c}\dfrac{\dfrac{\dfrac{\begin{matrix}\vdots\\ \circ\{X'\}, A_1\end{matrix}\quad \dfrac{\begin{matrix}\vdots\\ \circ\{X'\}, A_2\end{matrix}\quad \begin{matrix}\vdots\\ \overline{A}_1, \overline{A}_2, \bullet\{Y'\}\end{matrix}}{\overline{A}_1, \circ\{X'\}, \bullet\{Y'\}}\,cut}{\circ\{X'\}, \circ\{X'\}, \bullet\{Y'\}}\,cut}{\circ\{X'\}, \bullet\{Y'\}}\,ctr}{\circ\{\circ\{X'\}\}, Y'}\,rp\\ \vdots\\ \circ\{\circ\{X'\}\}, Y\end{array}$$

(5)

Figure 6.2: Some derivations in **SKt**.

(5) in Figure 6.2). Notice that the cuts on A_1 and A_2 introduced in the derivation above are on smaller formulae than A.

The above simplified explanation implicitly assumes that a uniform substitution of a formula (or formulae) in a derivation results in a well-formed derivation, and that the cut formulae are not contracted. The precise statement of the derivation substitution idea becomes more involved once these aspects are taken into account. The formal statement is given in the lemma below. We use the notation $\vdash_S X$ to denote that the sequent X is provable in the sequent calculus S. We write $\vdash_S \Pi : X$ when we want to be explicit about the particular derivation Π of X. The *cut rank* of an instance of cut is defined as usual, as the size of the cut formula. The cut rank of a derivation Π, denoted with $cr(\Pi)$, is the largest cut rank of the cut instances in Π (or zero, if there are no cuts in Π). Given a formula A, we denote with $|A|$ its size. Given a derivation Π, we denote with $|\Pi|$ its height, i.e., the length of a longest branch in the derivation tree of Π.

Lemma 6.1.8. *If* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, a$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[\bar{a}]$, *where* $k \geq 1$ *and both* Π_1 *and* Π_2 *are cut free, then there exists a cut free* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y]$.

Proof. By induction on $|\Pi_2|$. For the base cases, the non-trivial case is when Π_2 ends with *id* and $\bar{a}$ is active in the rule, i.e., $\Sigma^k[\bar{a}] = \Sigma_1^{k-1}[\bar{a}], \bar{a}, a$ and Π_2 is

$$\dfrac{}{\Sigma^k[\bar{a}] = \Sigma_1^{k-1}[\bar{a}], \bar{a}, a}\ id$$

Then we construct Π as follows:

$$\dfrac{\begin{matrix}\Pi_1\\ Y, a\end{matrix}}{\Sigma_1^{k-1}[Y], Y, a}\ wk$$

The inductive cases follow straightforwardly from the induction hypothesis.

Q.E.D.

Lemma 6.1.9. *Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, A$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : Y, B$ *and* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[\overline{A} \vee \overline{B}]$ *for some* $k \geq 1$, *and the cut ranks of* Π_1, Π_2 *and* Π *are smaller than* $|A \wedge B|$. *Then there exists a derivation* Π' *such that* $\vdash_{\mathbf{SKt}} \Pi' : \Sigma^k[Y]$ *and* $cr(\Pi) < |A \wedge B|$.

Proof. By induction on $|\Pi|$. Most cases are straightforward. The only non-trivial case is when $\overline{A} \vee \overline{B}$ is principal in the last rule of Π, i.e., Π is of the form

$$\dfrac{\begin{matrix}\Psi\\ \Sigma_1^{k-1}[\overline{A} \vee \overline{B}], \overline{A}, \overline{B}\end{matrix}}{\Sigma_1^{k-1}[\overline{A} \vee \overline{B}], \overline{A} \vee \overline{B}}\ \vee$$

By induction hypothesis, we have a cut-free derivation Ψ' such that

$$\vdash_{\mathbf{SKt}} \Psi' : \Sigma_1^{k-1}[Y], \overline{A}, \overline{B}.$$

The derivation Π' is constructed as follows:

$$\dfrac{\dfrac{\begin{matrix}\Pi_1\\ Y, A\end{matrix} \quad \dfrac{\begin{matrix}\Pi_2\\ Y, B\end{matrix} \quad \begin{matrix}\Psi'\\ \Sigma_1^{k-1}[Y], \overline{A}, \overline{B}\end{matrix}}{\Sigma_1^{k-1}[Y], \overline{A}, Y}\ cut}{\Sigma_1^{k-1}[Y], Y, Y}\ cut}{\Sigma_1^{k-1}[Y], Y}\ ctr$$

Q.E.D.

Lemma 6.1.10. *Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, A, B$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[\overline{A} \wedge \overline{B}]$, *for some* $k \geq 1$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|A \vee B|$. *Then there exists a derivation* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y]$ *and* $cr(\Pi) < |A \vee B|$.

Proof. This is proved analogously to Lemma 6.1.9.

Q.E.D.

To prove the next two lemmas, we use the following derived rules:

$$\frac{X, \circ\{Y_1\}, \circ\{Y_2\}}{X, \circ\{Y_1, Y_2\}}\, d1 \qquad \frac{X, \bullet\{Y_1\}, \bullet\{Y_2\}}{X, \bullet\{Y_1, Y_2\}}\, d2$$

These two rules are derivable using *rp*, *rf*, *ctr* and *wk*. The rule *d*1 is derived as follows (*d*2 is derived analogously):

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{X, \circ\{Y_1\}, \circ\{Y_2\}}{Y_2, \bullet\{X, \circ\{Y_1\}\}}\, rf}{Y_1, Y_2, \bullet\{X, \circ\{Y_1\}\}}\, wk}{X, \circ\{Y_1, Y_2\}, \circ\{Y_1\}}\, rp}{\bullet\{X, \circ\{Y_1, Y_2\}\}, Y_1}\, rf}{\bullet\{X, \circ\{Y_1, Y_2\}\}, Y_1, Y_2}\, wk}{X, \circ\{Y_1, Y_2\}, \circ\{Y_1, Y_2\}}\, rp}{X, \circ\{Y_1, Y_2\}}\, ctr$$

Lemma 6.1.11. *Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, \circ\{A\}$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[\Diamond\overline{A}]$, *for some* $k \geq 1$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\Box A|$. *Then there exists a derivation* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y]$ *and* $cr(\Pi) < |\Box A|$.

Proof. By induction on $|\Pi_2|$. The non-trivial case is when Π_2 ends with $\Diamond$ on $\Diamond\overline{A}$.

$$\cfrac{\begin{matrix}\Pi_2' \\ \Sigma_1^{k-1}[\Diamond\overline{A}], \circ\{X, \overline{A}\}\end{matrix}}{\Sigma_1^{k-1}[\Diamond\overline{A}], \circ\{X\}, \Diamond\overline{A}}\, \Diamond$$

By induction hypothesis we have $\vdash_{\mathbf{SKt}} \Pi' : \Sigma_1^{k-1}[Y], \circ\{X, \overline{A}\}$ such that $cr(\Pi') < |\Box A|$. The derivation Π is constructed as follows:

$$\cfrac{\cfrac{\cfrac{\cfrac{\begin{matrix}\Pi' \\ \Sigma_1^{k-1}[Y], \circ\{X, \overline{A}\}\end{matrix}}{\bullet\{\Sigma_1^{k-1}[Y]\}, X, \overline{A}}\, rf \qquad \cfrac{\begin{matrix}\Pi_1 \\ Y, \circ\{A\}\end{matrix}}{\bullet\{Y\}, A}\, rf}{\bullet\{\Sigma_1^{k-1}[Y]\}, \bullet\{Y\}, X}\, cut}{\bullet\{\Sigma_1^{k-1}[Y], Y\}, X}\, d2}{\Sigma_1^{k-1}[Y], \circ\{X\}, Y}\, rp$$

Q.E.D.

Lemma 6.1.12. *Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, \circ\{Y', A\}$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[\Box\overline{A}]$, *for some* $k \geq 1$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\Diamond A|$. *Then there exists* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y, \circ\{Y'\}]$ *and* $cr(\Pi) < |\Diamond A|$.

Proof. By induction on $|\Pi_2|$. The non-trivial case: Π_2 is

$$\dfrac{\begin{array}{c}\Pi_2'\\ \Sigma_1^{k-1}[\Box\overline{A}], \circ\{\overline{A}\}\end{array}}{\Sigma_1^{k-1}[\Box\overline{A}], \Box\overline{A}}\ \Box$$

By induction hypothesis, we have $\vdash_{\mathbf{SKt}} \Pi' : \Sigma_1^{k-1}[Y, \circ\{Y'\}], \circ\{\overline{A}\}$ for some Π' such that $cr(\Pi') < |\Diamond A|$. Then Π is constructed as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi'\\ \Sigma_1^{k-1}[Y, \circ\{Y'\}], \circ\{\overline{A}\}\end{array}}{\bullet\{\Sigma_1^{k-1}[Y, \circ\{Y'\}]\}, \overline{A}}\ rf \quad \dfrac{\begin{array}{c}\Pi_1\\ Y, \circ\{Y', A\}\end{array}}{\bullet\{Y\}, Y', A}\ rf}{\bullet\{\Sigma_1^{k-1}[Y, \circ\{Y'\}]\}, \bullet\{Y\}, Y'}\ cut}{\bullet\{\Sigma_1^{k-1}[Y, \circ\{Y'\}], Y\}, Y'}\ d2}{\Sigma_1^{k-1}[Y, \circ\{Y'\}], Y, \circ\{Y'\}}\ rp$$

Q.E.D.

Lemma 6.1.13. *Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, \bullet\{A\}$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[\blacklozenge\overline{A}]$, *for some* $k \geq 1$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\blacksquare A|$. *Then there exists a derivation* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y]$ *and* $cr(\Pi) < |\blacksquare A|$.

Proof. This is proved analogously to Lemma 6.1.11. Q.E.D.

Lemma 6.1.14. *Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, \bullet\{Y', A\}$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[\blacksquare\overline{A}]$, *for some* $k \geq 1$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\blacklozenge A|$. *Then there exists* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y, \bullet\{Y'\}]$ *and* $cr(\Pi) < |\blacklozenge A|$.

Proof. This is proved analogously to Lemma 6.1.12. Q.E.D.

Lemma 6.1.15. *Let* A *be a non-atomic formula. Suppose* $\vdash_{\mathbf{SKt}} \Pi_1 : Y, \overline{A}$ *and* $\vdash_{\mathbf{SKt}} \Pi_2 : \Sigma^k[A]$, *for some* $k \geq 1$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|A|$. *Then there exists a derivation* Π *such that* $\vdash_{\mathbf{SKt}} \Pi : \Sigma^k[Y]$ *and* $cr(\Pi) < |A|$.

Proof. By induction on the height of Π_2 and case analysis on A. The non-trivial case is when Π_2 ends with an introduction rule on A. That is, we have $\Sigma^k[A] = \Sigma_1^{k-1}[A], A$ for some context $\Sigma_1^{k-1}[]$. We show the cases where A is either $\Box B$, $\Diamond B$ or $B_1 \wedge B_2$.

- Suppose $A = \Box B$ and Π_2 is the following derivation:

$$\dfrac{\begin{array}{c}\Pi_2'\\ \Sigma_1^{k-1}[\Box B], \circ\{B\}\end{array}}{\Sigma_1^{k-1}[\Box B], \Box B}\ \Box$$

  By induction hypothesis, we have $\vdash_{\mathbf{SKt}} \Pi' : \Sigma_1^{k-1}[Y], \circ\{B\}$ and $cr(\Pi') < |A|$. Applying Lemma 6.1.11 to Π' and Π_1, we obtain $\vdash_{\mathbf{SKt}} \Pi : Y, \Sigma_1^{k-1}[Y] = \Sigma^k[Y]$ such that $cr(\Pi) < |\Box B|$.

- Suppose $A = \Diamond B$ and Π_2 is the following derivation:

$$\dfrac{\begin{array}{c}\Pi_2' \\ \Sigma_1^{k-1}[\Diamond B], \circ\{Y', B\}\end{array}}{\Sigma_1^{k-1}[\Diamond B], \circ\{Y'\}, \Diamond B}\ \Diamond$$

  By induction hypothesis, we have $\vdash_{\mathbf{SKt}} \Pi' : \Sigma_1^{k-1}[Y], \circ\{Y', B\}$. Applying Lemma 6.1.12 to Π' and Π_1, we obtain $\vdash_{\mathbf{SKt}} \Pi : Y, \Sigma_1^{k-1}[Y], \circ\{Y'\} = \Sigma_1^k[Y], \circ\{Y'\} = \Sigma^k[Y]$ such that $cr(\Pi) < |\Diamond B|$.

- Suppose $A = B_1 \wedge B_2$ and Π_2 is the following derivation:

$$\dfrac{\begin{array}{c}\Theta_1 \\ \Sigma_1^{k-1}[B_1 \wedge B_2], B_1\end{array} \qquad \begin{array}{c}\Theta_2 \\ \Sigma_1^{k-1}[B_1 \wedge B_2], B_2\end{array}}{\Sigma_1^{k-1}[B_1 \wedge B_2], B_1 \wedge B_2}\ \wedge$$

  By induction hypothesis, we have $\vdash_{\mathbf{SKt}} \Theta_1' : \Sigma_1^{k-1}[Y], B_1$ and $\vdash_{\mathbf{SKt}} \Theta_2' : \Sigma_1^{k-1}[Y], B_2$. Applying Lemma 6.1.9 to Θ_1' and Θ_2' and Π_1, we obtain $\vdash_{\mathbf{SKt}} \Pi : Y, \Sigma_1^{k-1}[Y] = \Sigma^k[Y]$ such that $cr(\Pi) < |B_1 \wedge B_2|$.

Q.E.D.

Theorem 6.1.16. *Cut elimination holds for* **SKt**.

Proof. We remove topmost cuts in succession. Let Π be a **SKt**-derivation with a topmost cut instance

$$\dfrac{\begin{array}{c}\Pi_1 \\ X, A\end{array} \quad \begin{array}{c}\Pi_2 \\ \overline{A}, Y\end{array}}{X, Y}\ cut$$

Note that Π_1 and Π_2 are both cut-free since this is a topmost instance in Π. We use induction on the size of A to eliminate this topmost instance of cut. If A is an atomic formula a then we obtain a cut-free derivation Π' of X, Y from applying Lemma 6.1.8 to Π_1 and Π_2.

If A is non-atomic then we apply Lemma 6.1.15 to Π_2 and Π_1 and obtain a derivation Π' of X, Y such that $cr(\Pi') < |A|$. By the induction hypothesis, we can remove all the cuts in Π' to get a cut-free derivation of X, Y. Q.E.D.

6.2 DKt: a contraction-free deep inference nested sequent calculus

We now consider another sequent calculus which uses deep inference, where rules can be applied directly to any node within a nested sequent. We call this calculus **DKt**, and give its inference rules in Figure 6.3. Note that there are no structural rules in **DKt**, and the contraction rule is absorbed into the logical rules. Notice also that, reading the logical rules bottom up, we keep the principal formulae in the premise. This is

$$\frac{}{\Sigma[a,\bar{a}]}\ id \qquad \frac{\Sigma[A\wedge B, A] \quad \Sigma[A\wedge B, B]}{\Sigma[A\wedge B]}\ \wedge \qquad \frac{\Sigma[A\vee B, A, B]}{\Sigma[A\vee B]}\ \vee$$

$$\frac{\Sigma[\blacksquare A, \bullet\{A\}]}{\Sigma[\blacksquare A]}\ \blacksquare \qquad \frac{\Sigma[\bullet\{Y, A\}, \blacklozenge A]}{\Sigma[\bullet\{Y\}, \blacklozenge A]}\ \blacklozenge_1 \qquad \frac{\Sigma[\circ\{Y, A\}, \Diamond A]}{\Sigma[\circ\{Y\}, \Diamond A]}\ \Diamond_1$$

$$\frac{\Sigma[\Box A, \circ\{A\}]}{\Sigma[\Box A]}\ \Box \qquad \frac{\Sigma[\circ\{Y, \blacklozenge A\}, A]}{\Sigma[\circ\{Y, \blacklozenge A\}]}\ \blacklozenge_2 \qquad \frac{\Sigma[\bullet\{Y, \Diamond A\}, A]}{\Sigma[\bullet\{Y, \Diamond A\}]}\ \Diamond_2$$

Figure 6.3: **DKt**: a contraction-free deep inference nested sequent calculus for Kt

actually not necessary for some rules (e.g., $\blacksquare$, $\wedge$, etc.), but this form of rule allows for a better accounting of formulae in our saturation-based proof search procedure (see Section 6.4).

The following intuitive observation about **DKt** rules will be useful later: Rules in **DKt** are characterized by propagations of formulae across different nodes in a nested sequent tree. The shape of the tree is not affected by these propagations, and the only change that can occur to the tree is the creation of new nodes (via the introduction rules $\blacksquare$ and $\Box$).

The calculus **DKt** corresponds to Kashima's **S2Kt** [73], but with the contraction rule absorbed into the logical rules. Kashima shows that **DKt** derivations can be encoded into **SKt**, essentially due to the display property of **SKt** (Proposition 6.1.3) which allows displaying and undisplaying of any node within a nested sequent. Kashima also shows that **DKt** is complete for tense logic, via semantic arguments. We prove a stronger result: every cut-free **SKt**-derivation can be transformed into a **DKt**-derivation, hence **DKt** is complete and cut is admissible in **DKt**.

To translate cut-free **SKt**-derivations into **DKt**-derivations, we show that all structural rules of **SKt** are height-preserving admissible in **DKt**, as stated next.

Lemma 6.2.1 (Admissibility of weakening)**.** *For any structures X and Y, if $\vdash_{\mathbf{DKt}} \Pi : \Sigma[X]$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \Sigma[X, Y]$ and $|\Pi'| = |\Pi|$.*

Proof. Straightforward by induction on $|\Pi|$. We give one case where Π ends with a propagation rule, and one case where Π ends with a logical rule; all other cases are analogous. In each of the following cases, we obtain Π'_1 from Π_1 using the induction hypothesis.

$$\frac{\begin{array}{c}\Pi_1\\ \Sigma[\circ\{X', \blacklozenge A\}, A]\end{array}}{\Sigma[\circ\{X', \blacklozenge A\}]}\ \blacklozenge_2 \quad \rightsquigarrow \quad \frac{\begin{array}{c}\Pi'_1\\ \Sigma[\circ\{X', \blacklozenge A\}, A, Y]\end{array}}{\Sigma[\circ\{X', \blacklozenge A\}, Y]}\ \blacklozenge_2$$

$$\frac{\begin{array}{c}\Pi_1\\ \Sigma[\Box A, \circ\{A\}]\end{array}}{\Sigma[\Box A]}\ \Box \quad \rightsquigarrow \quad \frac{\begin{array}{c}\Pi'_1\\ \Sigma[\Box A, \circ\{A\}, Y]\end{array}}{\Sigma[\Box A, Y]}\ \Box$$

Q.E.D.

Invertibility of our rules follows immediately, since or each of our rules, the premise is a superset of the conclusion, and weakening is height-preserving.

Lemma 6.2.2 (Invertibility)**.** *All* **DKt** *rules are invertible: if the conclusion is derivable, then each premise is derivable.*

The proofs for the following lemmas concern structural rules that change the shape of the tree of a nested sequent. Just as we saw in the corresponding lemmas in Chapter 5 on bi-intuitionistic logic (Lemmas 5.2.5 to 5.2.9), the only non-trivial cases are those that concern propagation of formulae across different nodes in a nested sequent. Again, we prove each lemma the lemma by induction on $|\Pi|$, and Π'_1 is obtained from induction hypothesis.

Lemma 6.2.3 (Admissibility of display postulate rp)**.** *For any X and Y, if $\vdash_{\mathbf{DKt}} \Pi : X, \bullet\{Y\}$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \circ\{X\}, Y$ and $|\Pi'| = |\Pi|$.*

Proof.

- Case when a formula is propagated from X to Y:

$$\dfrac{\begin{matrix}\Pi_1\\ X', \blacklozenge A, \bullet\{A, Y\}\end{matrix}}{X', \blacklozenge A, \bullet\{Y\}}\ \blacklozenge_1 \quad \leadsto \quad \dfrac{\begin{matrix}\Pi'_1\\ \circ\{X', \blacklozenge A\}, A, Y\end{matrix}}{\circ\{X', \blacklozenge A\}, Y}\ \blacklozenge_2$$

- Case when a formula is propagated from Y to X:

$$\dfrac{\begin{matrix}\Pi_1\\ X, \bullet\{Y', \Diamond A\}, A\end{matrix}}{X, \bullet\{Y', \Diamond A\}}\ \Diamond_2 \quad \leadsto \quad \dfrac{\begin{matrix}\Pi'_1\\ Y', \Diamond A, \circ\{X, A\}\end{matrix}}{Y', \Diamond A, \circ\{X\}}\ \Diamond_1$$

The cases involving other rules follow immediately from the induction hypothesis, since they do not move formulae between structures. Q.E.D.

Lemma 6.2.4 (Admissibility of display postulate rf)**.** *For any X and Y, if $\vdash_{\mathbf{DKt}} \Pi : X, \circ\{Y\}$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \bullet\{X\}, Y$ such that $|\Pi'| = |\Pi|$.*

Proof.

- Case when a formula is propagated from X to Y:

$$\dfrac{\begin{matrix}\Pi_1\\ X', \Diamond A, \circ\{A, Y\}\end{matrix}}{X', \Diamond A, \circ\{Y\}}\ \Diamond_1 \quad \leadsto \quad \dfrac{\begin{matrix}\Pi'_1\\ \bullet\{X', \Diamond A\}, A, Y\end{matrix}}{\bullet\{X', \Diamond A\}, Y}\ \Diamond_2$$

- Case when a formula is propagated from Y to X:

$$\dfrac{\begin{array}{c}\Pi_1\\ X, \circ\{Y', \blacklozenge A\}, A\end{array}}{X, \circ\{Y', \blacklozenge A\}}\,\blacklozenge_2 \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ Y', \blacklozenge A, \bullet\{X, A\}\end{array}}{Y', \blacklozenge A, \bullet\{X\}}\,\blacklozenge_1$$

The cases involving other rules follow immediately from the induction hypothesis, since they do not move formulae between structures. Q.E.D.

To show admissibility of contraction, we first need to show certain distributivity properties, stated in the following two lemmas, and admissibility of formula contraction.

Lemma 6.2.5. *For any Y_1 and Y_2, if $\vdash_{\mathbf{DKt}} \Pi : \Sigma[\circ\{Y_1\}, \circ\{Y_2\}]$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \Sigma[\circ\{Y_1, Y_2\}]$ and $|\Pi'| = |\Pi|$.*

Proof. By induction on $|\Pi|$. Again, the non-trivial cases are when a formula is propagated in or out of one of the structures. In each case, we obtain Π_1' using the induction hypothesis.

- Case when Π ends with $\Diamond_1$ that moves a formula into $\circ\{Y_1\}$.

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\Diamond A, \circ\{A, Y_1\}, \circ\{Y_2\}]\end{array}}{\Sigma'[\Diamond A, \circ\{Y_1\}, \circ\{Y_2\}]}\,\Diamond_1 \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma'[\Diamond A, \circ\{A, Y_1, Y_2\}]\end{array}}{\Sigma'[\Diamond A, \circ\{Y_1, Y_2\}]}\,\Diamond_1$$

- Case when Π ends with $\blacklozenge_2$ that moves a formula out from $\circ\{Y_1\}$.

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma[A, \circ\{\blacklozenge A, Y_1'\}, \circ\{Y_2\}]\end{array}}{\Sigma[\circ\{\blacklozenge A, Y_1'\}, \circ\{Y_2\}]}\,\blacklozenge_2 \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma[A, \circ\{\blacklozenge A, Y_1', Y_2\}]\end{array}}{\Sigma[\circ\{\blacklozenge A, Y_1', Y_2\}]}\,\blacklozenge_2$$

Q.E.D.

Lemma 6.2.6. *For any Y_1 and Y_2, if $\vdash_{\mathbf{DKt}} \Pi : \Sigma[\bullet\{Y_1\}, \bullet\{Y_2\}]$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \Sigma[\bullet\{Y_1, Y_2\}]$ and $|\Pi'| = |\Pi|$.*

Proof. Analogous to the proof of Lemma 6.2.5. Q.E.D.

Lemma 6.2.7. *If $\vdash_{\mathbf{DKt}} \Pi : \Sigma[A, A]$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \Sigma[A]$ and $|\Pi'| = |\Pi|$.*

Proof. Straightforward by induction on $|\Pi|$. We give one case where Π ends with a propagation rule, and one case where Π ends with a logical rule; all other cases are analogous. In each of the following cases, we obtain Π_1' from Π_1 using the induction hypothesis.

$$\begin{array}{c}\Pi_1\\ \Sigma[\circ\{X',\blacklozenge A,\blacklozenge A\},A]\\ \hline \Sigma[\circ\{X',\blacklozenge A,\blacklozenge A\}]\end{array}\,\blacklozenge_2 \quad\rightsquigarrow\quad \begin{array}{c}\Pi_1'\\ \Sigma[\circ\{X',\blacklozenge A\},A]\\ \hline \Sigma[\circ\{X',\blacklozenge A\}]\end{array}\,\blacklozenge_2$$

$$\begin{array}{c}\Pi_1\\ \Sigma[\Box A,\Box A,\circ\{A\}]\\ \hline \Sigma[\Box A,\Box A]\end{array}\,\Box \quad\rightsquigarrow\quad \begin{array}{c}\Pi_1'\\ \Sigma[\Box A,\circ\{A\}]\\ \hline \Sigma[\Box A]\end{array}\,\Box$$

Q.E.D.

Lemma 6.2.8 (Admissibility of general contraction). *For any Y, if $\vdash_{\mathbf{DKt}} \Pi : \Sigma[Y,Y]$ then there exists Π' such that $\vdash_{\mathbf{DKt}} \Pi' : \Sigma[Y]$ and $|\Pi'| = |\Pi|$.*

Proof. By induction on the size of Y.

- If Y is a singleton set containing one formula, then the lemma follows immediately from Lemma 6.2.7.
- The other cases follow from the induction hypothesis and Lemma 6.2.5 and Lemma 6.2.6. Consider, for instance, the case where $Y = \circ\{Y'\}$. Then by Lemma 6.2.5 we have a derivation Ψ such that

  $$\vdash_{\mathbf{DKt}} \Psi : \Sigma[\circ\{Y',Y'\}]$$

  and $|\Psi| = |\Pi|$. Since Y' is of a smaller size than $\circ\{Y'\}$, we can apply the induction hypothesis to Ψ and obtain a derivation Π' such that

  $$\vdash_{\mathbf{DKt}} \Pi' : \Sigma[\circ\{Y'\}]$$

  and $|\Pi'| = |\Pi|$.

Q.E.D.

Lemma 6.2.9. *For every sequent X, if $\vdash_{\mathbf{DKt}} X$ then $\vdash_{\mathbf{SKt}} X$.*

Proof. We use the display property of **SKt** (Proposition 6.1.3) to simulate the deep-inference rules of **DKt**. We show here the derivations for the rules $\Diamond_1$ and $\Diamond_2$ (the other cases are analogous), where the dashed line indicates an application of Proposition 6.1.3:

$$\begin{array}{cl} \Sigma[\circ\{Y,A\},\Diamond A] & \\ \hdashline Y',\circ\{Y,A\},\Diamond A & \text{Prop. 6.1.3}\\ \hline Y',\circ\{Y\},\Diamond A,\Diamond A & \Diamond\\ \hline Y',\circ\{Y\},\Diamond A & ctr\\ \hdashline \Sigma[\circ\{Y\},\Diamond A] & \text{Prop. 6.1.3} \end{array} \qquad \begin{array}{cl} \Sigma[\circ\{Y,\blacklozenge A\},A] & \\ \hdashline Y',\circ\{Y,\blacklozenge A\},A & \text{Prop. 6.1.3}\\ \hline \bullet\{Y',A\},Y,\blacklozenge A & rf\\ \hline \bullet\{Y'\},\blacklozenge A,Y,\blacklozenge A & \blacklozenge\\ \hline \bullet\{Y'\},Y,\blacklozenge A & ctr\\ \hline Y',\circ\{Y,\blacklozenge A\} & rp\\ \hdashline \Sigma[\circ\{Y,\blacklozenge A\}] & \text{Prop. 6.1.3} \end{array}$$

Q.E.D.

Lemma 6.2.10. *For every sequent X, if $\vdash_{\mathbf{SKt}} \Pi : X$ then $\vdash_{\mathbf{DKt}} \Pi' : X$.*

Proof. By induction on Π. We illustrate one case where Π ends with a logical rule application, and one case where Π ends with a structural rule application. The other cases are analogous and use admissibility of the structural rules of **SKt** in **DKt** (Lemmas 6.2.1 to 6.2.8). In each case, we obtain Π_1' from the induction hypothesis, and a dashed line indicates the application of a lemma.

$$\dfrac{\begin{matrix}\Pi_1\\ X, \bullet\{A\}\end{matrix}}{X, \blacksquare A}\,\blacksquare \qquad \rightsquigarrow \qquad \dfrac{\dfrac{\begin{matrix}\Pi_1'\\ X, \bullet\{A\}\end{matrix}}{X, \blacksquare A, \bullet\{A\}}\text{ Lemma 6.2.1}}{X, \blacksquare A}\,\blacksquare$$

$$\dfrac{\begin{matrix}\Pi_1\\ X, \circ\{Y\}\end{matrix}}{\bullet\{X\}, Y}\,rf \qquad \rightsquigarrow \qquad \dfrac{\begin{matrix}\Pi_1'\\ X, \circ\{Y\}\end{matrix}}{\bullet\{X\}, Y}\text{ Lemma 6.2.4}$$

Q.E.D.

Theorem 6.2.11 (Equivalence)**.** *For every sequent X, $\vdash_{\mathbf{SKt}} X$ if and only if $\vdash_{\mathbf{DKt}} X$.*

Proof. By Lemmas 6.2.9 and 6.2.10. Q.E.D.

A consequence of Theorem 6.2.11 is that the general contraction rule in **SKt** can be replaced by formula contraction. This can be proved as follows: take a cut-free derivation in **SKt**, translate it to **DKt** and then translate it back to **SKt**. Since general contraction is admissible in **DKt**, and since the translation from **DKt** to **SKt** does not use general contraction (only formula contraction), we can effectively replace the general contraction in **SKt** with formula contraction.

6.3 Sequent calculi for some extensions of tense logic

We now consider extensions of tense logic with some modal axioms. We show that, for each extension, there is a shallow inference calculus that modularly extends **SKt** for which cut elimination holds. By modular extension we mean that the rules of the extended calculi are the rules of **SKt** plus some structural rules that are derived directly from the modal axioms. We then show that for each extension, there is also a corresponding deep inference calculus which is equivalent to the shallow one. Again, as with **DKt**, the rules for the deep calculi are characterized by propagations of formulae across different nodes in the nested sequents.

However, the design of the rules for the deep calculus is not as modular as its shallow counterpart, since it needs to take into account the closure of the axioms. This is not surprising: Brünnler's nested sequent calculi for modal logics [20] have similar properties. That is, the versions of his calculi which use modal propagation rules have terminating proof search procedures but lack modularity. These calculi are very similar to our deep calculi (but we consider tense rather than modal logic). On the other

$$\frac{\Sigma[\blacklozenge A, A]}{\Sigma[\blacklozenge A]}\,T_a \qquad \frac{\Sigma[\blacklozenge A, \bullet\{\blacklozenge A, Y\}]}{\Sigma[\blacklozenge A, \bullet\{Y\}]}\,4_a \qquad \frac{\Sigma[\lozenge A, \circ\{\lozenge A, Y\}]}{\Sigma[\lozenge A, \circ\{Y\}]}\,4_c$$

$$\frac{\Sigma[\lozenge A, A]}{\Sigma[\lozenge A]}\,T_b \qquad \frac{\Sigma[\circ\{Y, \blacklozenge A\}, \blacklozenge A]}{\Sigma[\circ\{Y, \blacklozenge A\}]}\,4_b \qquad \frac{\Sigma[\bullet\{Y, \lozenge A\}, \lozenge A]}{\Sigma[\bullet\{Y, \lozenge A\}]}\,4_d$$

Figure 6.4: Additional propagation rules for **DS4**

hand, the versions of Brünnler's calculi that use structural rules for capturing extensions of modal logic do not yield proof search procedures, but are modular. These calculi are similar to our shallow calculi.

Cut elimination holds for all the extensions discussed in the following. Their proofs are omitted as they are a straightforward adaptation of the cut elimination proof for **SKt**. This is because the proof substitution technique used for cut elimination in **SKt** relies on rule applications being invariant under formula substitution. More precisely, all the additional structural rules that we shall consider have the following property: If there is an instance of a structural rule ρ (below left) then instantiating the occurrences of A in the multi-context Σ_1 and Σ_2 with any structure Y yields a valid instance of ρ (below right):

$$\frac{\Sigma_2^k[A]}{\Sigma_1^k[A]}\,\rho \qquad\qquad \frac{\Sigma_2^k[Y]}{\Sigma_1^k[Y]}\,\rho.$$

Hence the proof substitution technique for cut elimination goes through essentially unchanged for the extended logic. This property of the structural rules is similar to Belnap's condition (C6) for cut elimination for display logics [12].

A *primitive axiom* is an axiom of the form $A \to B$ where both A and B are built using propositional variables, $\wedge$, $\vee$, $\lozenge$, and $\blacklozenge$. Kracht [75] shows that any extension of tense logic with primitive axioms has a display calculus which enjoys cut elimination. He shows that any such axiom can be turned into a left structural rule. The axioms we consider next are contrapositives of primitive axioms, so Kracht's translation from axioms to structural rules in our formalism gives right structural rules. We illustrate here a few cases of primitive axioms for which one can also get corresponding deep inference nested sequent calculi.

6.3.1 Modal tense logic `Kt.S4`

Consider an extension of **SKt** with the following axioms:

$$T : \Box A \to A \quad \blacksquare A \to A \qquad 4 : \Box A \to \Box\Box A \quad \blacksquare A \to \blacksquare\blacksquare A.$$

These axioms translate into the following structural rules, whose soundness is immediately derivable from the axioms:

$$\frac{X, \bullet\{Y\}}{X, Y}\, T_p \qquad \frac{X, \circ\{Y\}}{X, Y}\, T_f \qquad \frac{X, \bullet\{Y\}}{X, \bullet\{\bullet\{Y\}\}}\, 4_p \qquad \frac{X, \circ\{Y\}}{X, \circ\{\circ\{Y\}\}}\, 4_f$$

Definition 6.3.1 (Calculus **SS4**). *The calculus* **SS4** *is* **SKt** *plus T_p, T_f, 4_p and 4_f.*

Theorem 6.3.2. *Cut elimination holds for* **SS4**.

Definition 6.3.3 (Calculus **DS4**). *The calculus* **DS4** *is* **DKt** *plus the propagation rules given in Figure 6.4.*

Some of the modal rules of **DS4** coincide with Brünnler's rules for T and 4 [17; 20]. As the following lemma shows, the rules of **DS4** are derivable in **SS4**.

Lemma 6.3.4. *Every rule of* **DS4** *is derivable in* **SS4**.

Proof. We show here derivations of T_a, 4_a and 4_b; the others are analogous. In each case, dashed lines indicate multiple applications of the residuation rules rp and/or rf, according to Proposition 6.1.3.

- Rule T_a:

$$\begin{array}{cl}
\Sigma[\blacklozenge A, A] & \\
\hdashline
X, \blacklozenge A, A & \text{Prop. 6.1.3} \\
\hline
\bullet\{\ \}, X, \blacklozenge A, A & wk \\
\hline
\circ\{X, \blacklozenge A, A\} & rp \\
\hline
\circ\{X, \blacklozenge A\}, \blacklozenge A & \blacklozenge \\
\hline
X, \blacklozenge A, \blacklozenge A & T_f \\
\hline
X, \blacklozenge A & ctr \\
\hdashline
\Sigma[\blacklozenge A] & \text{Prop. 6.1.3}
\end{array}$$

- Rule 4_a:

$$\dfrac{\begin{array}{cl}
\hline
\circ\{\}, A, \overline{A} & id \\
\hline
\bullet\{A, \overline{A}\} & rf \\
\hline
\blacklozenge A, \bullet\{\overline{A}\} & \blacklozenge \\
\hline
\blacklozenge A, \bullet\{\bullet\{\overline{A}\}\} & 4_p \\
\hline
\circ\{\blacklozenge A\}, \bullet\{\overline{A}\} & rp \\
\hline
\circ\{\blacklozenge A\}, \blacksquare\overline{A} & \Box \\
\hline
\blacklozenge A, \bullet\{\blacksquare\overline{A}\} & rf \\
\hline
\blacklozenge A, \blacksquare\blacksquare\overline{A} & \blacksquare
\end{array}
\qquad
\begin{array}{cl}
\Sigma[\blacklozenge A, \bullet\{\blacklozenge A, Y\}] & \\
\hdashline
X, \blacklozenge A, \bullet\{\blacklozenge A, Y\} & \text{Prop. 6.1.3} \\
\hline
X, \blacklozenge A, \blacklozenge\blacklozenge A, \bullet\{Y\} & \blacklozenge
\end{array}}
{\begin{array}{cl}
X, \blacklozenge A, \blacklozenge A, \bullet\{Y\} & cut \\
\hline
X, \blacklozenge A, \bullet\{Y\} & ctr \\
\hdashline
\Sigma[\blacklozenge A, \bullet\{Y\}] & \text{Prop. 6.1.3}
\end{array}}$$

- Rule 4_b: note that the following derivation uses the derived rule 4_a:

$$\begin{array}{cl}
\Sigma[\blacklozenge A, \circ\{Y, \blacklozenge A\}] & \\
\hdashline
X, \blacklozenge A, \circ\{Y, \blacklozenge A\} & \text{Prop. 6.1.3} \\
\hline
\bullet\{X, \blacklozenge A\}, Y, \blacklozenge A & rf \\
\hline
\bullet\{X\}, Y, \blacklozenge A & 4_a \\
\hline
X, \circ\{Y, \blacklozenge A\} & rp \\
\hdashline
\Sigma[\circ\{Y, \blacklozenge A\}] & \text{Prop. 6.1.3}
\end{array}$$

Q.E.D.

To prove the equivalence of **SS4** and **DS4**, we first need to prove the analogs of Lemmas 6.2.1 to 6.2.8, stated below.

Lemma 6.3.5. *The following hold:*

1. *(Admissibility of weakening) For any structures X and Y, if $\vdash_{\mathbf{DS4}} \Pi : \Sigma[X]$ then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \Sigma[X, Y]$ and $|\Pi'| = |\Pi|$.*

2. *(Admissibility of rp) For any X and Y, if $\vdash_{\mathbf{DS4}} \Pi : X, \bullet\{Y\}$ then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \circ\{X\}, Y$ and $|\Pi'| = |\Pi|$.*

3. *(Admissibility of rf) For any X and Y, if $\vdash_{\mathbf{DS4}} \Pi : X, \circ\{Y\}$ then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \bullet\{X\}, Y$ such that $|\Pi'| = |\Pi|$.*

4. *(Admissibility of general contraction) For any Y, if $\vdash_{\mathbf{DS4}} \Pi : \Sigma[Y, Y]$ then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \Sigma[Y]$ and $|\Pi'| = |\Pi|$.*

Proof. The proof of these statements are a straightforward extension of the proofs of Lemmas 6.2.1 to 6.2.8. This is because:

- the extra propagation rules T_a and T_b are not context-dependent, so they are not affected by changes in the structure of the context, and

- the extra propagation rules 4_a to 4_d have the same structures as the rules $\Diamond_1$ to $\blacklozenge_2$ and hence are treated analogously.

Q.E.D.

Additionally, we need to show that the structural rules for the axioms T and 4 are also admissible in **DS4**, which we do in the following four lemmas. The principle behind the proofs of admissibility for these structural rules is the same as for the rules rp and rf in Section 6.2. That is, the non-trivial cases we need to consider are those that concern propagation of formulae across structures affected by the structural rules. Unless stated otherwise, all lemmas in this section are proved by induction on $|\Pi|$, and Π_1' is obtained from Π_1 using the induction hypothesis. We label a dashed line with the lemma used to obtain the conclusion from the premise.

Lemma 6.3.6 (Admissibility of T_f). *For all Y, if $\vdash_{\mathbf{DS4}} \Pi : \Sigma[\circ\{Y\}]$, then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \Sigma[Y]$.*

Proof. The non-trivial case is when Π ends with a diamond-rule that moves a formula in or out of $\circ\{Y\}$.

- Case when Π ends with $\Diamond_1$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\circ\{Y, A\}, \Diamond A]\end{array}}{\Sigma'[\circ\{Y\}, \Diamond A]}\ \Diamond_1 \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma'[Y, A, \Diamond A]\end{array}}{\Sigma'[Y, \Diamond A]}\ T_b$$

- Case when Π ends with $\blacklozenge_2$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma[\circ\{Y', \blacklozenge A\}, A]\end{array}}{\Sigma[\circ\{Y', \blacklozenge A\}]}\ \blacklozenge_2 \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma[Y', \blacklozenge A, A]\end{array}}{\Sigma[Y', \blacklozenge A]}\ T_a$$

- Case when Π ends with 4_c:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\Diamond A, \circ\{\Diamond A, Y\}]\end{array}}{\Sigma'[\Diamond A, \circ\{Y\}]}\ 4_c \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \Sigma'[\Diamond A, \Diamond A, Y]\end{array}}{\Sigma'[\Diamond A, Y]}\ \text{Lemma 6.3.5 (4)}$$

The other cases involving axiom 4 can be done analogously.

Q.E.D.

Lemma 6.3.7 (Admissibility of T_p). *Suppose $\vdash_{\mathbf{DS4}} \Pi : \Sigma[\bullet\{Y\}]$. Then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \Sigma[Y]$.*

Proof. Analogous to the proof of Lemma 6.3.6. Q.E.D.

Lemma 6.3.8 (Admissibility of 4_f). *Suppose $\vdash_{\mathbf{DS4}} \Pi : \Sigma[\circ\{Y\}]$. Then there exists Π' such that $\vdash_{\mathbf{DS4}} \Pi' : \Sigma[\circ\{\circ\{Y\}\}]$.*

Proof. The non-trivial cases are when a diamond formula moves in or out of $\circ\{Y\}$.

- Case when Π ends with $\blacklozenge_2$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma[\circ\{Y', \blacklozenge A\}, A]\end{array}}{\Sigma[\circ\{Y', \blacklozenge A\}]}\ \blacklozenge_2 \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma[\circ\{\circ\{Y', \blacklozenge A\}\}, A]\end{array}}{\Sigma[\circ\{\circ\{Y', \blacklozenge A\}, \blacklozenge A\}, A]}\ \text{Lemma 6.3.5 (1)}}{\Sigma[\circ\{\circ\{Y', \blacklozenge A\}, \blacklozenge A\}]}\ \blacklozenge_2}{\Sigma[\circ\{\circ\{Y', \blacklozenge A\}\}]}\ 4_b$$

$$\frac{\Sigma[\blacklozenge A, \circ\{\blacklozenge A, Y\}]}{\Sigma[\blacklozenge A, \circ\{Y\}]}\, 5_a \qquad \frac{\Sigma[\circ\{Y, \lozenge A\}, \lozenge A]}{\Sigma[\circ\{Y, \lozenge A\}]}\, 5_b$$

$$\frac{\Sigma[\lozenge A, \bullet\{\lozenge A, Y\}]}{\Sigma[\lozenge A, \bullet\{Y\}]}\, 5_c \qquad \frac{\Sigma[\bullet\{Y, \blacklozenge A\}, \blacklozenge A]}{\Sigma[\bullet\{Y, \blacklozenge A\}]}\, 5_d$$

Figure 6.5: Additional propagation rules for **DS5**

- Case when Π ends with $\lozenge_1$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\circ\{Y, A\}, \lozenge A]\end{array}}{\Sigma'[\circ\{Y\}, \lozenge A]}\,\lozenge_1 \quad \rightsquigarrow \quad \dfrac{\dfrac{\begin{array}{c}\Pi'_1\\ \Sigma'[\circ\{\circ\{Y, A\}\}, \lozenge A]\end{array}}{\Sigma'[\circ\{\circ\{Y, A\}, \lozenge A\}, \lozenge A]}\,\text{Lemma 6.3.5 (1)}}{\dfrac{\Sigma'[\circ\{\circ\{Y\}, \lozenge A\}, \lozenge A]}{\Sigma'[\circ\{\circ\{Y\}\}, \lozenge A]}\,4_c}\,\lozenge_1$$

The other cases can be proved analogously. Q.E.D.

Lemma 6.3.9 (Admissibility of 4_p). *Suppose* $\vdash_{\mathbf{DS4}} \Pi : \Sigma[\bullet\{Y\}]$. *Then there exists* Π' *such that* $\vdash_{\mathbf{DS4}} \Pi' : \Sigma[\bullet\{\bullet\{Y\}\}]$.

Proof. Analogous to the proof of Lemma 6.3.8. Q.E.D.

Theorem 6.3.10. *For every* X, *we have* $\vdash_{\mathbf{SS4}} X$ *if and only if* $\vdash_{\mathbf{DS4}} X$.

Proof. Left-to-right direction: a corollary of Lemmas 6.3.5 to 6.3.9, using the method of Lemma 6.2.10. Right-to-left direction: Lemma 6.3.4. Q.E.D.

6.3.2 Modal tense logic S5

We can obtain S5 from **SS4** by collapsing $\Box$ and $\blacksquare$. That is, the symmetry axiom $B : A \to \Box\lozenge A$ splits into two axioms given below, which translate straightforwardly into two structural rules.

$$B1 : \blacksquare A \to \Box A \qquad \frac{X, \bullet\{Y\}}{X, \circ\{Y\}}\, B_1 \qquad B2 : \Box A \to \blacksquare A \qquad \frac{X, \circ\{Y\}}{X, \bullet\{Y\}}\, B_2$$

Definition 6.3.11 (Calculus **SS5**). *The calculus* **SS5** *is* **SS4** *plus the rules* B_1 *and* B_2.

Theorem 6.3.12. *Cut elimination holds for* **SS5**.

Definition 6.3.13 (Calculus **DS5**). *The calculus* **DS5** *is* **DS4** *plus the propagation rules given in Figure 6.5.*

Lemma 6.3.14. *Every rule of* **DS5** *is derivable in* **SS5**.

Proof. The following are derivations of the rules 5_a and 5_b; note that propagation rules 4_a and 4_d are derivable in **SS5** by Lemma 6.3.4, since **SS5** is a superset of **SS4**.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Sigma[\blacklozenge A, \circ\{\blacklozenge A, Y\}]}{\blacklozenge A, \circ\{\blacklozenge A, Y\}, X}\;\text{Prop. 6.1.3}}{\blacklozenge A, \bullet\{\blacklozenge A, Y\}, X}\;B_2}{\blacklozenge A, \bullet\{Y\}, X}\;4_a}{\blacklozenge A, \circ\{Y\}, X}\;B_1}{\Sigma[\blacklozenge A, \circ\{Y\}]}\;\text{Prop. 6.1.3}
\qquad
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Sigma[\circ\{Y, \Diamond A\}, \Diamond A]}{\circ\{Y, \Diamond A\}, \Diamond A, X}\;\text{Prop. 6.1.3}}{\bullet\{Y, \Diamond A\}, \Diamond A, X}\;B_2}{\bullet\{Y, \Diamond A\}, X}\;4_d}{\circ\{Y, \Diamond A\}, X}\;B_1}{\Sigma[\Diamond A, \circ\{Y\}]}\;\text{Prop. 6.1.3}$$

The derivations of rules 5_c and 5_d are analogous and use the derived propagation rules 4_c and 4_b respectively. Q.E.D.

To prove the equivalence of **SS5** and **DS5**, we again first need to prove the analogs of Lemmas 6.2.1 to 6.2.8, stated below.

Lemma 6.3.15. *The following hold:*

1. *(Admissibility of weakening) For any structures X and Y, if $\vdash_{\mathbf{DS5}} \Pi : \Sigma[X]$ then there exists Π' such that $\vdash_{\mathbf{DS5}} \Pi' : \Sigma[X, Y]$ and $|\Pi'| = |\Pi|$.*

2. *(Admissibility of rp) For any X and Y, if $\vdash_{\mathbf{DS5}} \Pi : X, \bullet\{Y\}$ then there exists Π' such that $\vdash_{\mathbf{DS5}} \Pi' : \circ\{X\}, Y$ and $|\Pi'| = |\Pi|$.*

3. *(Admissibility of rf) For any X and Y, if $\vdash_{\mathbf{DS5}} \Pi : X, \circ\{Y\}$ then there exists Π' such that $\vdash_{\mathbf{DS5}} \Pi' : \bullet\{X\}, Y$ such that $|\Pi'| = |\Pi|$.*

4. *(Admissibility of general contraction) For any Y, if $\vdash_{\mathbf{DS5}} \Pi : \Sigma[Y, Y]$ then there exists Π' such that $\vdash_{\mathbf{DS5}} \Pi' : \Sigma[Y]$ and $|\Pi'| = |\Pi|$.*

Proof. Similarly to Lemma 6.3.5, the proof of these statements are a straightforward extension of the proofs of Lemmas 6.2.1 to 6.2.8. This is because the extra propagation rules 5_a to 5_d have the same structures as the rules $\Diamond_1$ to $\blacklozenge_2$ and hence are treated analogously. Q.E.D.

We now prove the admissibility of the rules corresponding to the axioms of **SS4** and structural rules B_1 and B_2. Note that **DS5** captures $S5 = KT4B$ rather than $S5 = KT45$.

Lemma 6.3.16 (Admissibility of B_1). *Suppose $\vdash_{\mathbf{DS5}} \Pi : \Sigma[\circ\{Y\}]$. Then there exists Π' such that $\vdash_{\mathbf{DS5}} \Pi' : \Sigma[\bullet\{Y\}]$.*

Proof. The non-trivial cases are when Π ends with a diamond-rule that moves a formula in or out of $\circ\{Y\}$.

- Case when Π ends with $\Diamond_1$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\circ\{Y',A\},\Diamond A]\end{array}}{\Sigma'[\circ\{Y'\},\Diamond A]}\,\Diamond_1 \quad\rightsquigarrow\quad \dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi'_1\\ \Sigma'[\bullet\{Y',A\},\Diamond A]\end{array}}{\Sigma'[\bullet\{Y',A,\Diamond A\},\Diamond A]}\,\text{Lemma 6.3.15 (1)}}{\Sigma'[\bullet\{Y',\Diamond A\},\Diamond A]}\,T_b}{\Sigma'[\bullet\{Y'\},\Diamond A]}\,5_c$$

- Case when Π ends with $\blacklozenge_2$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\circ\{Y',\blacklozenge A\},A]\end{array}}{\Sigma'[\circ\{Y',\blacklozenge A\}]}\,\blacklozenge_2 \quad\rightsquigarrow\quad \dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi'_1\\ \Sigma'[\bullet\{Y',\blacklozenge A\},A]\end{array}}{\Sigma'[\bullet\{Y',\blacklozenge A\},A,\blacklozenge A]}\,\text{Lemma 6.3.15 (1)}}{\Sigma'[\bullet\{Y',\blacklozenge A\},\blacklozenge A]}\,T_a}{\Sigma'[\bullet\{Y',\blacklozenge A\}]}\,5_d$$

- Case when Π ends with 4_b:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma[\circ\{Y',\blacklozenge A\},\blacklozenge A]\end{array}}{\Sigma[\circ\{Y',\blacklozenge A\}]}\,4_b \quad\rightsquigarrow\quad \dfrac{\begin{array}{c}\Pi'_1\\ \Sigma[\bullet\{Y',\blacklozenge A\},\blacklozenge A]\end{array}}{\Sigma[\bullet\{Y',\blacklozenge A\}]}\,5_d$$

- Case when Π ends with 4_c:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\Diamond A,\circ\{\Diamond A,Y\}]\end{array}}{\Sigma'[\Diamond A,\circ\{Y\}]}\,4_c \quad\rightsquigarrow\quad \dfrac{\begin{array}{c}\Pi'_1\\ \Sigma[\Diamond A,\bullet\{\Diamond A,Y\}]\end{array}}{\Sigma'[\Diamond A,\bullet\{Y\}]}\,5_c$$

- Case when Π ends with 5_a:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\blacklozenge A,\circ\{\blacklozenge A,Y'\}]\end{array}}{\Sigma'[\blacklozenge A,\circ\{Y'\}]}\,5_a \quad\rightsquigarrow\quad \dfrac{\begin{array}{c}\Pi'_1\\ \Sigma'[\blacklozenge A,\bullet\{\blacklozenge A,Y'\}]\end{array}}{\Sigma'[\blacklozenge A,\bullet\{Y'\}]}\,4_a$$

- Case when Π ends with 5_b:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma'[\circ\{Y',\Diamond A\},\Diamond A]\end{array}}{\Sigma'[\circ\{Y',\Diamond A\}]}\,5_b \quad\rightsquigarrow\quad \dfrac{\begin{array}{c}\Pi'_1\\ \Sigma'[\bullet\{Y',\Diamond A\},\Diamond A]\end{array}}{\Sigma'[\bullet\{Y',\Diamond A\}}\,4_d$$

The other cases are analogous. Q.E.D.

Lemma 6.3.17 (Admissibility of B_2)**.** *Suppose* $\vdash_{\mathbf{DS5}} \Pi : \Sigma[\bullet\{Y\}]$. *Then there exists* Π' *such that* $\vdash_{\mathbf{DS5}} \Pi' : \Sigma[\circ\{Y\}]$.

```
Function Prove (Sequent $\Xi$) : Bool

  1. Let $T = tree(\Xi)$
  2. If the id rule is applicable to any node in $T$, return True
  3. Else if there is some node $\Theta \in T$ that is not saturated
     (a) If $A \lor B \in \Theta$ and $A \notin \Theta$ or $B \notin \Theta$ then let $\Xi_1$ be the premise of the $\lor$ rule
         applied to $A \lor B \in \Theta$. Return $Prove(\Xi_1)$.
     (b) If $A \land B \in \Theta$ and $A \notin \Theta$ and $B \notin \Theta$ then let $\Xi_1$ and $\Xi_2$ be the premises
         of the $\land$ rule applied to $A \land B \in \Theta$. Return True iff $Prove(\Xi_1) = True$ and
         $Prove(\Xi_2) = True$.
  4. Else if there is some node $\Theta$ that is not propagated
     (a) Let $\rho$ be the rule corresponding to the requirement of Definition 6.4.3 that
         is not met, and let $\Xi_1$ be the premise of $\rho$. Return $Prove(\Xi_1)$.
  5. Else if there is some node $\Theta \in T$ that is not realised, i.e. some $B = \Box A$ ($B = \blacksquare A$)
     is not realised
     (a) Let $\Xi_1$ be the premise of the $\Box$ ($\blacksquare$) rule applied to $B \in \Theta$. Return $Prove(\Xi_1)$.
  6. Else return False
```

Figure 6.6: A proof search strategy for **DKt**

Proof. Analogous to the proof of Lemma 6.3.16. Q.E.D.

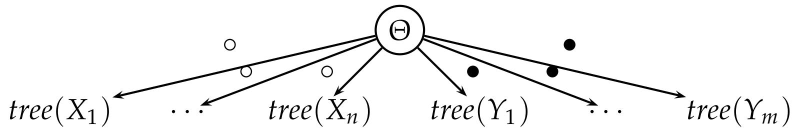
Theorem 6.3.18. *For every X, we have $\vdash_{\mathbf{SS5}} X$ if and only if $\vdash_{\mathbf{DS5}} X$.*

Proof. Left-to-right direction: a corollary of Lemmas 6.3.15 to 6.3.17, using the method of Lemma 6.2.10. Right-to-left direction: Lemma 6.3.14. Q.E.D.

6.4 Proof search

We can devise a terminating proof search strategy for our deep inference nested sequent calculus **DKt**. While traditional tableaux methods operate on a single node at a time, our proof search strategies will consider the whole tree. Following Kashima, first we define a mapping from sequents to trees.

A *node* is a set of formulae. A *tree* is a node with 0 or more children, where each child is a tree, and each child is labelled as either a $\circ$-child, or a $\bullet$-child. Given a sequent $\Xi = \Theta, \circ\{X_1\}, \cdots, \circ\{X_n\}, \bullet\{Y_1\}, \cdots, \bullet\{Y_m\}$, where Θ is a set of formulae and $n \geq 0$ and $m \geq 0$, the tree $tree(\Xi)$ represented by Ξ is:



Definition 6.4.1. *A set of formulae Θ is saturated iff it satisfies:*

1. *If $A \vee B \in \Theta$ then $A \in \Theta$ and $B \in \Theta$.*
2. *If $A \wedge B \in \Theta$ then $A \in \Theta$ or $B \in \Theta$.*

Definition 6.4.2. *Given a tree T and a node $\Theta \in T$, a formula $\Box A \in \Theta$ ($\blacksquare A \in \Theta$) is realised iff there exists a $\circ$-child ($\bullet$-child) X of Θ in T with $A \in X$.*

6.4.1 Proof Search in DKt

Figure 6.6 gives a proof search strategy for **DKt**. The application of a rule deep inside a sequent can be viewed as focusing on a particular node of the tree. The rules of **DKt** can then be viewed as operations on the tree encoded in the sequent. In particular, Step 3 saturates a node locally, Step 4 propagates $\Diamond$ ($\blacklozenge$) prefixed formulae between neighbouring nodes, and Step 5 appends new nodes to the tree.

Definition 6.4.3. *Given a tree T and a node $\Theta \in T$, we say Θ is propagated iff:*

$\Diamond_1$: *for every $\Diamond A \in \Theta$ and for every $\circ$-child X of Θ, we have $A \in X$*

$\blacklozenge_1$: *for every $\blacklozenge A \in \Theta$ and for every $\bullet$-child X of Θ, we have $A \in X$*

$\Diamond_2$: *for every $\bullet$-child X of Θ and for every $\Diamond A \in X$, we have $A \in \Theta$*

$\blacklozenge_2$: *for every $\circ$-child X of Θ and for every $\blacklozenge A \in X$, we have $A \in \Theta$*

We will now show that *Prove* is complete for tense logic. As was the case for $\mathbf{DBiInt_1}$ in Chapter 5, we show the contrapositive: if *Prove* returns false for some sequent Ξ, then Ξ is not valid. Again, we use a purely syntactic method via the sound and complete calculus **DKt**. The following auxiliary lemma shows that if *Prove* returns false for some sequent Ξ, there is no **DKt** derivation of that sequent, and uses essentially the same ideas as Lemma 6.4.4.

Lemma 6.4.4. *Let Ξ be a sequent such that every node in $tree(\Xi)$ is saturated, realised and propagated. Then Ξ is not derivable in* **DBiInt**.

Proof. Similar to the proof of Lemma 5.3.5, using Lemmas 6.2.5, 6.2.6 and 6.2.7. We give the most difficult case as an example.

We prove the statement of the lemma by contradiction. That is, we assume every node in $tree(\Xi)$ is saturated, realised and propagated, and that Ξ is derivable in **DBiInt**. Then there exists a shortest derivation Π of Ξ. We now consider all the rule instances that Π could end with, and in each case show that there is an even shorter **DBiInt** derivation of Ξ, therefore contradicting our assumption that Π was the shortest derivation.

- Π cannot end with the *id* rule, since every node in $tree(\Xi)$ is saturated.
- Suppose Π ends with the $\Box$ rule. Since every node in $tree(\Xi)$ is realised, we have $\Xi = \Sigma[X_1, \Box A, \circ\{A, Y_1\}]$ for some $\Sigma[]$ and the last rule of Π applies to that particular occurrence of $\Box A$ in the context $\Sigma[]$. Then Π must be of the form:

  $$\dfrac{\begin{matrix}\Pi_1\\ \Sigma[X_1, \Box A, \circ\{A, Y_1\}, \circ\{A\}]\end{matrix}}{\Sigma[X_1, \Box A, \circ\{A, Y_1\}]}\ \Box$$

  By the distribution Lemma 6.2.5, there is a **DKt** derivation Π_2 of

  $$\Sigma[X_1, \Box A, \circ\{A, A, Y_1\}]$$

  such that $|\Pi_2| = |\Pi_1|$. Then applying Lemma 6.2.7 to Π_2 gives us a **DKt** derivation Π_3 of

  $$\Sigma[X_1, \Box A, \circ\{A, Y_1\}]$$

  such that $|\Pi_3| = |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\Box$.
- The cases for other rules are very similar to the proof of Lemma 5.3.5.

Since Π cannot end with any of the rules of **DKt**, this obviously contradicts the assumption that it is a derivation in **DKt**. Therefore Ξ is not derivable in **DKt**. Q.E.D.

Theorem 6.4.5. *For any X, Prove(X) = true if and only if* $\vdash_{\mathbf{DKt}} \Pi : X$.

Proof.

- Left-to-right: obvious, since every step of *Prove* is a backwards application of a **DKt** rule.
- Right-to-left: we show that if $Prove(X)$ returns False then X is not derivable in **DKt**. Since each rule of **DKt** is invertible (Lemma 6.2.2), Steps 3 to 5 of *Prove* preserve provability of the original sequent. If $Prove(X)$ returns False, this can only be the case if Step 6 is reached, i.e., the systematic bottom-up applications of the rules of **DKt** produce a sequent such that every node in the tree of the sequent is saturated, realised, and propagated. By Lemma 6.4.4, such a sequent would not be derivable in **DKt**, and since all other steps of *Prove* preserve derivability, it follows that X is not derivable either in **DKt**.

Q.E.D.

The degree of a formula is the maximum number of nested modalities:

$$\begin{array}{rcl} deg(p) & = & 0 \\ deg(A\#B) & = & max(deg(A), deg(B)) \text{ for } \# \in \{\wedge, \vee\} \\ deg(\#A) & = & 1 + deg(A) \text{ for } \# \in \{\Box, \Diamond, \blacksquare, \blacklozenge\}. \end{array}$$

The degree of a set of formulae is the maximum degree over all its members. We write $sf(A)$ for the subformulae of A, and define the set of subformulae of a set Θ as $sf(\Theta) = \bigcup_{A \in \Theta} sf(A)$. For a sequent Ξ we define $sf(\Xi)$ as below:

$$\begin{array}{rcl} \Xi & = & \Theta, \circ\{X_1\}, \cdots, \circ\{X_n\}, \bullet\{Y_1\}, \cdots, \bullet\{Y_m\} \\ sf(\Xi) & = & sf(\Theta) \cup sf(X_1) \cup \cdots \cup sf(X_n) \cup sf(Y_1) \cup \cdots \cup sf(Y_m). \end{array}$$

Theorem 6.4.6. *For any set of* **DKt**-*formulae* Γ, *Prove*(Γ) *terminates.*

Proof. Let $m = |sf(\Gamma)|$ and $d = deg(sf(\Gamma)) \leq m$. To show that *Prove* terminates, we will argue about the tree $T = tree(\Xi)$, where Ξ is the parameter to the most recent recursive call to *Prove*. That is, initially $T = tree(\Gamma)$.

The saturation process for each node in T is bounded by m. Therefore after at most m moves at each node, Step 3 is no longer applicable to this node.

T is finitely branching, since new nodes are only created for unrealised box formulae. Therefore after at most m moves at each node, Step 5 is no longer applicable to this node. The depth of T is bounded by d, since each node $\Theta \in T$ at distance k from the root of T has $degree(\Theta) \leq d - k$.

Since $\Diamond$- and $\blacklozenge$-prefixed formulae are only propagated to nodes that do not already contain these formulae, after at most m propagation moves into each node, Step 4 is no longer applicable to this node. Q.E.D.

Putting it all together: bi-intuitionistic tense logic

Having studied bi-intuitionistic logic and tense logic separately, a natural question to ask is whether our results are applicable to the combination of these two logics. We address this question in this chapter: specifically, we develop shallow and deep inference nested sequent calculi for bi-intuitionistic tense logic.

First, we introduce and motivate our study of bi-intuitionistic tense logic, starting with intuitionistic modal logic. Recall that traditional modal and tense logics have a classical basis; that is, they are conservative extensions of classical logic and the box and diamond connectives are interdefinable: $\Box A = \neg\Diamond\neg A$ and $\blacksquare A = \neg\blacklozenge\neg A$. On the other hand, intuitionistic modal/tense logics (IM/TLs) use intuitionistic logic as a basis. There is a long history behind IM/TLs: Simpson [104] gives a comprehensive survey of IML and Ewald [39] presents Hilbert calculi for ITL and various extensions. In addition to Hilbert calculi with algebraic, topological or relational semantics, sequent and natural deduction calculi for IM/TLs have also been developed [84; 1; 92; 31; 42]. Extending these sequent calculi with "converse" modalities like $\blacklozenge$ and $\blacksquare$ causes cut-elimination to fail as it does for the bi-logics we have encountered so far. Simpson's labelled sequent calculus [104] can help but it is not purely syntactic since it encodes the Kripke semantics, as discussed further in Section 8.4.

In addition to being conservative extensions of intuitionistic logic, IM/TLs also have a number of other interesting features. Indeed, Simpson [104] states the following properties as requirements of a "good" IML; we extend his requirements to IM/TL:

1. IM/TL is conservative over IPL.
2. IM/TL contains all substitution instances of theorems of IPL and is closed under modus ponens.
3. The addition of the schema $A \vee \neg A$ to IM/TL yields a standard classical modal/tense logic.
4. If $A \vee B$ is a theorem of IM/TL then either A is a theorem or B is a theorem.

5. $\Box$ and $\Diamond$ are independent in IM/TL; $\blacksquare$ and $\blacklozenge$ are independent in IM/TL.

6. There is an intuitionistically comprehensible explanation of the meaning of the modalities, relative to which IM/TL is sound and complete.

While requirements 1 to 5 are non-controversial, there is a large variety of semantics available for IM/TLs, and as a result not all IM/TLs meet requirement 6. As discussed by Simpson [104], there are a number of possibilities of obtaining a Kripke semantics for IM/TL:

- We could start by simply combining the semantics of intuitionistic logic with that of modal/tense logic, leading us to consider structures of the form $\langle W, \leq, R, V \rangle$, where W is a set of worlds, $\leq$ is the intuitionistic partial order on worlds in W, R is the modal/tense reachability relation and V is the valuation.

- While the interpretation of the intuitionistic connectives is straightforward, how should we interpret the modalities? If we use the usual satisfaction clauses for modalities, then the intuitionistic persistence property no longer holds. In order to maintain the persistence property, we need to either build it into the satisfaction clauses for modalities, or impose conditions on models to ensure persistence.

- Should there be a relationship between $\leq$ and R? It might be considered natural for the most basic IM/TL to permit arbitrary frames; on the other hand, it turns out that certain relationships between $\leq$ and R are necessary for obtaining the full persistence property.

Ewald gives a philosophical justification of his semantics for ITL. He argues that *"we should view our models as partially-ordered set of 'states-of-knowledge,' which we think of as belonging to a tense-logician who is studying a set of times. Within each state-of-knowledge there is a set of times and a temporal ordering. As the tense logician moves to a greater state-of-knowledge, he retains all the information he had in lesser states-of-knowledge." [39]* In other words, Ewald is advocating that not only formulae should be persisted, but the temporal ordering itself should be persisted as well: if we have uRv at a certain time point w, then for all time points w' such that $w' \geq w$, we should have $u'Rv'$, where $u' \geq u$ and $v' \geq v$.

We will now present bi-intuitionistic tense logic, which extends bi-intuitionistic logic with tense formulae just as IML extends intuitionistic logic with modal formulae, and show that our logic satisfies many of the requirements given above. Let `BiKt` be the bi-intuitionistic tense logic obtained by extending `BiInt` with two pairs of adjoint modalities $(\Diamond, \blacksquare)$ and $(\blacklozenge, \Box)$, with no explicit relationship between the modalities of the same colour, namely, $(\Diamond, \Box)$ and $(\blacklozenge, \blacksquare)$. The logic `BiKt` enjoys various desirable properties:

- **Conservativity:** it is a conservative extension of both intuitionistic logic `Int` and dual intuitionistic logic `DualInt`;

$$
\begin{array}{lcllcl}
\tau^-(\emptyset) & = & \top & \tau^+(\emptyset) & = & \bot \\
\tau^-(A) & = & A & \tau^+(A) & = & A \\
\tau^-(X,Y) & = & \tau^-(X) \wedge \tau^-(Y) & \tau^+(X,Y) & = & \tau^+(X) \vee \tau^+(Y) \\
\tau^-(X \triangleright Y) & = & \tau^-(X) \prec \tau^+(Y) & \tau^+(X \triangleright Y) & = & \tau^-(X) \to \tau^+(Y) \\
\tau^-(\circ X) & = & \Diamond \tau^-(X) & \tau^+(\circ X) & = & \Box \tau^+(X) \\
\tau^-(\bullet X) & = & \blacklozenge \tau^-(X) & \tau^+(\bullet X) & = & \blacksquare \tau^+(X)
\end{array}
$$

$$
\tau(X \triangleright Y) \quad = \quad \tau^-(X) \to \tau^+(Y) \quad = \quad \tau(X \Rightarrow Y)
$$

Figure 7.1: Formula translation of nested sequents

- **Classical Collapse:** it collapses to classical tense logic by the addition of four structural rules;
- **Disjunction Property:** If $A \vee B$ is a theorem not containing $\prec$ then A is a theorem or B is a theorem;
- **Dual Disjunction Property:** If $A \wedge B$ is a counter-theorem not containing $\to$ then so is A or B;
- **Independent $\Diamond$ and $\Box$:** there is no *a priori* relationship between these connectives

Following the methodology of the previous three chapters, we begin in Section 7.1 with a shallow inference calculus **LBiKt**, which is a merger of two sub-calculi for `BiInt` and `Kt` derived from Belnap's inherently modular display logic. **LBiKt** has syntactic cut-elimination as we show in Section 7.2, but it is ill-suited for backward proof search. We then give a deep inference calculus **DBiKt** which is complete with respect to the cut-free fragment of **LBiKt** as we show in Section 7.3. Importantly, all the structural rules of **LBiKt** are admissible in **DBiKt**, allowing us to give a terminating decision procedure in Section 7.4. To complete the picture, we also give a Kripke semantics for `BiKt` based upon three relations $\leq$, $R_\Diamond$ and $R_\Box$ in Section 7.5, and show that our calculi are sound and complete w.r.t. this semantics. Finally, in Section 7.6 we show how we can capture existing calculi for IM/TLs, as well as obtain classical versions of the logics considered here.

Note. A previous version of some of the results of this chapter has been published in [56].

7.1 Nested sequent calculi

The **formulae** of `BiKt` are built from a set *Atoms* of atomic formulae via the grammar below, with $p \in Atoms$:

$$
\begin{array}{lcl}
A & ::= & p \mid \top \mid \bot \mid A \to A \mid A \prec A \mid A \wedge A \mid A \vee A \mid \\
 & & \Box A \mid \Diamond A \mid \blacksquare A \mid \blacklozenge A.
\end{array}
$$

A structure is defined by the following grammar, where A is a `BiKt` formula:

$$X := \emptyset \mid A \mid (X, X) \mid X \triangleright X \mid \circ(X \triangleright X) \mid \bullet(X \triangleright X).$$

The structural connective "," is associative and commutative and $\emptyset$ is its unit. We always consider structures modulo these equivalences. To reduce parentheses, we assume that "," binds tighter than "$\triangleright$". Thus, we write $X, Y \triangleright Z$ to mean $(X, Y) \triangleright Z$.

If X and Y are structures, then $X \triangleright Y$ is a *nested deep sequent*, and $X \Rightarrow Y$ is a *nested shallow sequent*. Thus we generalise Kashima's sequents for classical tense logics [73] to two-sided sequents as required for the bi-intuitionistic aspects of `BiKt`. Figure 7.1 shows the formula-translation of nested sequents. On both sides of the sequent, $\circ$ is interpreted as a white (modal) operator and $\bullet$ as a black (tense) operator.

Similarly to the previous chapters, a *context* is a structure with a hole or a place-holder $[]$. Contexts are ranged over by $\Sigma[]$. We write $\Sigma[X]$ for the structure obtained by filling the hole $[]$ in the context $\Sigma[]$ with a structure X. A *simple* context is defined via:

$$\Sigma[] ::= [] \mid \Sigma[], (Y) \mid (Y), \Sigma[]$$

Intuitively, the hole in a simple context is never under the scope of $\triangleright$. The hole in a simple context is of *neutral* polarity.

Positive and negative contexts are defined inductively as follows:

- If $\Sigma[]$ is a simple context then the following are negative contexts:
  - $\Sigma[] \triangleright Y$
  - $\circ(\Sigma[] \triangleright Y)$
  - $\bullet(\Sigma[] \triangleright Y)$
- If $\Sigma[]$ is a simple context then the following are positive contexts:
  - $Y \triangleright \Sigma[]$
  - $\circ(Y \triangleright \Sigma[])$
  - $\bullet(Y \triangleright \Sigma[])$
- If $\Sigma[]$ is a positive/negative context then so are the following:
  - $(\Sigma[], Y)$ and $(Y, \Sigma[])$
  - $\Sigma[] \triangleright Y$ and $Y \triangleright \Sigma[]$
  - $\circ(\Sigma[] \triangleright Y)$ and $\circ(Y \triangleright \Sigma[])$
  - $\bullet(\Sigma[] \triangleright Y)$ and $\bullet(Y \triangleright \Sigma[])$

The hole in a negative context has negative polarity, and the hole in a positive context has positive polarity. We write $\Sigma^{-}[]$ to indicate that $\Sigma[]$ is a negative context and $\Sigma^{+}[]$ to indicate that it is a positive context. Note again that our definition of polarities is non-traditional since further nesting within $\triangleright$ *does not* change polarity. Thus, due to the constructive nature of `BiKt`, our calculi for `BiKt` bear more similarities to the calculi for `BiInt` we presented in Chapters 4 and 5 than they do to the calculi for `Kt` from Chapter 6.

Example 7.1.1. *The context* $[], (X \triangleright Y)$ *is a simple context but* $\bullet(([], X) \triangleright Y)$ *is not. Both* $\bullet(([], X) \triangleright Y)$ *and* $X \triangleright (([], Y) \triangleright Z)$ *are negative contexts. Both* $\bullet(W \triangleright ([], Z))$ *and* $X \triangleright ([], Y)$ *are positive contexts.*

We now present the two nested sequent calculi that we will use in the rest of the paper: a shallow inference calculus **LBiKt** and a deep inference calculus **DBiKt**.

Figure 7.2 gives the rules of the shallow inference calculus **LBiKt**. Central to this calculus is the idea that inference rules can only be applied to formulae at the top level of nested sequents, and the structural *residuation rules* s_L, s_R, $\triangleright_L$, $\triangleright_R$, $rp_\circ$ and $rp_\bullet$ are used to bring the required sub-structures to the top level. As in previous chapters, we use double lines to indicate that the rules $rp_\circ$ and $rp_\bullet$ may be applied both top-to-bottom and bottom-to-top.

LBiKt can be seen as a merger of two calculi: the **LBiInt** calculus presented in Chapter 4 for the intuitionistic connectives, and the display calculus [51] for the tense connectives. However, note that our $rp_\circ$ and $rp_\bullet$ are more intricate than in the case of pure display logic. Namely, we have incorporated some aspects of the residuation rules for $\triangleright$ connectives in $rp_\circ$ and $rp_\bullet$. This allows us to prove the admissibility of $rp_\circ$ and $rp_\bullet$ more easily in the deep inference calculus presented shortly.

Note that we use $\circ$ and $\bullet$ as structural proxies for the non-residuated pairs $(\Diamond, \Box)$ and $(\blacklozenge, \blacksquare)$ respectively, whereas Wansing [116] uses only one $\bullet$ as a structural proxy for the residuated pair $(\blacklozenge, \Box)$ and recovers $(\Diamond, \blacksquare)$ via classical negation, while Goré [51] uses $\circ$ and $\bullet$ as structural proxies for the residuated pairs $(\Diamond, \blacksquare)$ and $(\blacklozenge, \Box)$ respectively. As we shall see later, our choice allows us to retain the modal fragment $(\Diamond, \Box)$ by simply eliding all rules that contain "black" operators from our deep sequent calculus.

Figure 7.3 gives the rules of the deep inference calculus **DBiKt**. Here the inference rules can be applied at any level of the nested sequent, indicated by the use of contexts. Notably, there are no residuation rules; indeed one of the goals of our paper is to show that the residuation rules of **LBiKt** can be simulated by deep inference and propagation rules in **DBiKt**. Another feature of **DBiKt** is the use of polarities in defining contexts to which rules are applicable. For example, the premise of the $\Box_{L1}$ rule denotes a negative context Σ which itself contains a formula A and a $\bullet$-structure, such that the $\bullet$-structure contains $\Box A$.

DBiKt achieves the goal of merging the **DBiInt** calculus presented in Chapter 5 and a two-sided version of the **DKt** calculus presented in Chapter 6. Similarly to the shallow inference case, combining the calculi is not entirely straightforward. Although the propagation rules for $\triangleright$-structures remain the same as in the `BiInt` case, the propagation rules for $\circ$- and $\bullet$-structures are not as simple as in the **DKt** calculus. Since `BiKt` is constructive, there is no direct relationship between $\Box$ and $\Diamond$, or $\blacksquare$ and $\blacklozenge$. As a result, propagation rules like $\blacksquare_{L2}$ need to involve the $\triangleright$ structural connective so they can refer to both sides of the nested sequent.

Identity and logical constants:

$$\frac{}{X, A \Rightarrow A, Y}\ id \qquad \frac{}{X, \bot \Rightarrow Y}\ \bot_L \qquad \frac{}{X \Rightarrow \top, Y}\ \top_R$$

Structural rules:

$$\frac{X \Rightarrow Z}{X, Y \Rightarrow Z}\ w_L \qquad \frac{X \Rightarrow Z}{X \Rightarrow Y, Z}\ w_R \qquad \frac{X, Y, Y \Rightarrow Z}{X, Y \Rightarrow Z}\ c_L \qquad \frac{X \Rightarrow Y, Y, Z}{X \Rightarrow Y, Z}\ c_R$$

$$\frac{(X_1 \rhd Y_1), X_2 \Rightarrow Y_2}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_L \qquad \frac{X_1 \Rightarrow Y_1, (X_2 \rhd Y_2)}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R$$

$$\frac{X_2 \Rightarrow Y_2, Y_1}{(X_2 \rhd Y_2) \Rightarrow Y_1}\ \rhd_L \qquad \frac{X_1, X_2 \Rightarrow Y_2}{X_1 \Rightarrow (X_2 \rhd Y_2)}\ \rhd_R$$

$$\frac{X \Rightarrow Y, \circ(W \rhd Z)}{\bullet(X \rhd Y), W \Rightarrow Z}\ rp_\circ \qquad \frac{X \Rightarrow Y, \bullet(W \rhd Z)}{\circ(X \rhd Y), W \Rightarrow Z}\ rp_\bullet$$

$$\frac{X_1 \Rightarrow Y_1, A \qquad A, X_2 \Rightarrow Y_2}{X_1, X_2 \Rightarrow Y_1, Y_2}\ cut$$

Logical rules:

$$\frac{X, B_i \Rightarrow Y}{X, B_1 \wedge B_2 \Rightarrow Y}\ \wedge_L\ i \in \{1,2\} \qquad \frac{X \Rightarrow A, Y \qquad X \Rightarrow B, Y}{X \Rightarrow A \wedge B, Y}\ \wedge_R$$

$$\frac{X, A \Rightarrow Y \qquad X, B \Rightarrow Y}{X, A \vee B \Rightarrow Y}\ \vee_L \qquad \frac{X \Rightarrow B_i, Y}{X \Rightarrow B_1 \vee B_2, Y}\ \vee_R\ i \in \{1,2\}$$

$$\frac{X \Rightarrow A, Y \qquad X, B \Rightarrow Y}{X, A \to B \Rightarrow Y}\ \to_L \qquad \frac{X, A \Rightarrow B}{X \Rightarrow A \to B, Y}\ \to_R$$

$$\frac{A \Rightarrow B, Y}{X, A \prec B \Rightarrow Y}\ \prec_L \qquad \frac{X \Rightarrow A, Y \qquad X, B \Rightarrow Y}{X \Rightarrow A \prec B, Y}\ \prec_R$$

$$\frac{A \Rightarrow (X \rhd Y)}{\Box A \Rightarrow \circ(X \rhd Y)}\ \Box_L \qquad \frac{X \Rightarrow \circ(\emptyset \rhd A)}{X \Rightarrow \Box A}\ \Box_R \qquad \frac{A \Rightarrow (X \rhd Y)}{\blacksquare A \Rightarrow \bullet(X \rhd Y)}\ \blacksquare_L \qquad \frac{X \Rightarrow \bullet(\emptyset \rhd A)}{X \Rightarrow \blacksquare A}\ \blacksquare_R$$

$$\frac{\circ(A \rhd \emptyset) \Rightarrow X}{\Diamond A \Rightarrow X}\ \Diamond_L \qquad \frac{(X \rhd Y) \Rightarrow A}{\circ(X \rhd Y) \Rightarrow \Diamond A}\ \Diamond_R \qquad \frac{\bullet(A \rhd \emptyset) \Rightarrow X}{\blacklozenge A \Rightarrow X}\ \blacklozenge_L \qquad \frac{(X \rhd Y) \Rightarrow A}{\bullet(X \rhd Y) \Rightarrow \blacklozenge A}\ \blacklozenge_R$$

Figure 7.2: **LBiKt**: a shallow inference nested sequent calculus for `BiKt`

Identity and logical constants:

$$\frac{}{\Sigma[X, A \rhd A, Y]}\,id \qquad \frac{}{\Sigma[\bot, X \rhd Y]}\,\bot_L \qquad \frac{}{\Sigma[X \rhd \top, Y]}\,\top_R$$

Propagation rules:

$$\frac{\Sigma[A, (A, X \rhd Y), W \rhd Z]}{\Sigma[(A, X \rhd Y), W \rhd Z]}\,\rhd_{L1} \qquad \frac{\Sigma[W \rhd Z, (X \rhd Y, A), A]}{\Sigma[W \rhd Z, (X \rhd Y, A)]}\,\rhd_{R1}$$

$$\frac{\Sigma[X, A \rhd W, (A, Y \rhd Z)]}{\Sigma[X, A \rhd W, (Y \rhd Z)]}\,\rhd_{L2} \qquad \frac{\Sigma[(X \rhd Y, A), W \rhd A, Z]}{\Sigma[(X \rhd Y), W \rhd A, Z]}\,\rhd_{R2}$$

$$\frac{\Sigma^-[A, \bullet(\Box A, X \rhd Y)]}{\Sigma^-[\bullet(\Box A, X \rhd Y)]}\,\Box_{L1} \qquad \frac{\Sigma^+[A, \bullet(Y \rhd \Diamond A, X)]}{\Sigma^+[\bullet(Y \rhd \Diamond A, X)]}\,\Diamond_{R1}$$

$$\frac{\Sigma^-[A, \circ(\blacksquare A, X \rhd Y)]}{\Sigma^-[\circ(\blacksquare A, X \rhd Y)]}\,\blacksquare_{L1} \qquad \frac{\Sigma^+[A, \circ(Y \rhd \blacklozenge A, X)]}{\Sigma^+[\circ(Y \rhd \blacklozenge A, X)]}\,\blacklozenge_{R1}$$

$$\frac{\Sigma[\Box A, X \rhd \circ(A, Y \rhd Z), W]}{\Sigma[\Box A, X \rhd \circ(Y \rhd Z), W]}\,\Box_{L2} \qquad \frac{\Sigma[X, \circ(Y \rhd Z, A) \rhd W, \Diamond A]}{\Sigma[X, \circ(Y \rhd Z) \rhd W, \Diamond A]}\,\Diamond_{R2}$$

$$\frac{\Sigma[\blacksquare A, X \rhd \bullet(A, Y \rhd Z), W]}{\Sigma[\blacksquare A, X \rhd \bullet(Y \rhd Z), W]}\,\blacksquare_{L2} \qquad \frac{\Sigma[X, \bullet(Y \rhd Z, A) \rhd W, \blacklozenge A]}{\Sigma[X, \bullet(Y \rhd Z) \rhd W, \blacklozenge A]}\,\blacklozenge_{R2}$$

Logical rules:

$$\frac{\Sigma^-[A \vee B, A] \qquad \Sigma^-[A \vee B, B]}{\Sigma^-[A \vee B]}\,\vee_L \qquad \frac{\Sigma^+[A \vee B, A, B]}{\Sigma^+[A \vee B]}\,\vee_R$$

$$\frac{\Sigma^-[A \wedge B, A, B]}{\Sigma^-[A \wedge B]}\,\wedge_L \qquad \frac{\Sigma^+[A \wedge B, A] \qquad \Sigma^+[A \wedge B, B]}{\Sigma^+[A \wedge B]}\,\wedge_R$$

$$\frac{\Sigma^-[A \mathrel{-\!\!<} B, (A \rhd B)]}{\Sigma^-[A \mathrel{-\!\!<} B]}\,\mathrel{-\!\!<}_L \qquad \frac{\Sigma^+[A \to B, (A \rhd B)]}{\Sigma^+[A \to B]}\,\to_R$$

$$\frac{\Sigma[X, A \to B \rhd A, Y] \qquad \Sigma[X, A \to B, B \rhd Y]}{\Sigma[X, A \to B \rhd Y]}\,\to_L$$

$$\frac{\Sigma[X \rhd Y, A \mathrel{-\!\!<} B, A] \qquad \Sigma[X, B \rhd Y, A \mathrel{-\!\!<} B]}{\Sigma[X \rhd Y, A \mathrel{-\!\!<} B]}\,\mathrel{-\!\!<}_R$$

$$\frac{\Sigma^-[\Diamond A, \circ(A \rhd \emptyset)]}{\Sigma^-[\Diamond A]}\,\Diamond_L \qquad \frac{\Sigma^+[\Box A, \circ(\emptyset \rhd A)]}{\Sigma^+[\Box A]}\,\Box_R$$

$$\frac{\Sigma^-[\blacklozenge A, \bullet(A \rhd \emptyset)]}{\Sigma^-[\blacklozenge A]}\,\blacklozenge_L \qquad \frac{\Sigma^+[\blacksquare A, \bullet(\emptyset \rhd A)]}{\Sigma^+[\blacksquare A]}\,\blacksquare_R$$

Figure 7.3: **DBiKt**: a deep inference nested sequent calculus for `BiKt`

Notice that **LBiKt** is a modular extension of the calculus **LBiInt** for bi-intuitionistic logic that we presented in Chapter 4. That is, if we dropped all the rules involving tense formulae and tense structures from **LBiKt**, we would get exactly **LBiInt**. Similarly, **DBiKt** is a modular extension of **DBiInt** from Chapter 5.

As in previous chapters, we write $\vdash_{\mathbf{LBiKt}} \Pi : X \Rightarrow Y$ when Π is derivation of the shallow sequent $X \Rightarrow Y$ in **LBiKt**, and $\vdash_{\mathbf{DBiKt}} \Pi : X \rhd Y$ when Π is a derivation of the sequent $X \rhd Y$ in **DBiKt**. In either calculus, the height $|\Pi|$ of a derivation Π is the number of sequents on the longest branch.

Example 7.1.2. *Below we derive Ewald's axiom 9 for IK_t [39] in* **LBiKt** *and* **DBiKt**. *The* **LBiKt**-*derivation on the left read bottom-up brings the required sub-structure $\blacklozenge A$ to the top-level using the residuation rule $rp_\circ$ and applies $\blacklozenge_R$ backward. The* **DBiKt**-*derivation on the right instead applies $\Box_R$ deeply, and propagates the required formula to the appropriate sub-structure using $\blacklozenge_{R1}$. Note that $\blacklozenge_{R1}$ and all other propagation rules contain contraction.*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \Rightarrow A}\,id}{\bullet(A \rhd \emptyset) \Rightarrow \blacklozenge A}\,\blacklozenge_R}{A \Rightarrow \circ(\emptyset \rhd \blacklozenge A)}\,rp_\circ}{A \Rightarrow \Box\blacklozenge A}\,\Box_R}{\Rightarrow A \to \Box\blacklozenge A}\,\to_R \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\rhd A \to \Box\blacklozenge A, (A \rhd A, \Box\blacklozenge A, \circ(\emptyset \rhd \blacklozenge A))}\,id}{\rhd A \to \Box\blacklozenge A, (A \rhd \Box\blacklozenge A, \circ(\emptyset \rhd \blacklozenge A))}\,\blacklozenge_{R1}}{\rhd A \to \Box\blacklozenge A, (A \rhd \Box\blacklozenge A)}\,\Box_R}{\rhd A \to \Box\blacklozenge A}\,\to_R$$

We start by defining four derived "display" rules in **LBiKt**. The following two rules are easily derivable using s_L, s_R, $\rhd_L$ and $\rhd_R$:

$$\dfrac{(X_1 \rhd X_2) \Rightarrow Y}{X_1 \Rightarrow X_2, Y}\,rp^{\rhd}_{L} \qquad \dfrac{X_1 \Rightarrow (X_2 \rhd Y)}{X_1, X_2 \Rightarrow Y}\,rp^{\rhd}_{R}$$

The following two residuation rules are easily derivable using $rp^{\rhd}_{L}$, $rp^{\rhd}_{R}$, $rp_\circ$ and $rp_\bullet$:

$$\dfrac{\circ(X \rhd Y) \Rightarrow (Z \rhd Y)}{(X \rhd Y) \Rightarrow \bullet(Z \rhd Y)}\,rp'_\bullet \qquad \dfrac{\bullet(X \rhd Y) \Rightarrow (Z \rhd Y)}{(X \rhd Y) \Rightarrow \circ(Z \rhd Y)}\,rp'_\circ$$

Display property. A (deep or shallow) nested sequent can be seen as a tree of traditional sequents. The structural rules of **LBiKt** allow shuffling of structures to display/un-display a particular node in the tree, so inference rules can be applied to it. This is similar to the display property in traditional display calculi, where any substructure can be displayed and un-displayed. We state the display property of **LBiKt** more precisely in subsequent lemmas. Let $DP = \{rp^{\rhd}_{L}, rp^{\rhd}_{R}, rp_\circ, rp_\bullet, rp'_\circ, rp'_\bullet\}$ and let DP-derivable mean "derivable using rules only from DP".

The proofs for the following lemmas are all extensions of the proofs for lemmas 4.1.3, 4.1.4 and 4.1.5. The differences concern the cases for contexts under the scope of a $\circ$ or $\bullet$ structural connective, so we give these cases only.

Lemma 7.1.3 (Display property for simple contexts). *Let $\Sigma[]$ be a simple context. Let X be a structure and p a propositional variable not occurring in X nor $\Sigma[]$. Then there exist structures Y and Z such that:*

1. *$Y \Rightarrow p$ is DP-derivable from $X \Rightarrow \Sigma[p]$ and*
2. *$p \Rightarrow Z$ is DP-derivable from $\Sigma[p] \Rightarrow X$.*

Proof. Similar from the proof Lemma 4.1.3. Q.E.D.

Lemma 7.1.4 (Display property for positive contexts)**.** *Let $\Sigma[]$ be a positive context. Let X be a structure and p a propositional variable not occurring in X nor $\Sigma[]$. Then there exist structures Y and Z such that:*

1. *$Y \Rightarrow p$ is DP-derivable from $X \Rightarrow \Sigma[p]$, and*
2. *$Z \Rightarrow p$ is DP-derivable from $\Sigma[p] \Rightarrow X$*

Proof. We prove both statements by simultaneous induction on the size of the context $\Sigma[]$. We give the cases when $\Sigma[] = \bullet(\Sigma_1[] \triangleright W)$ or $\Sigma[] = \bullet(W \triangleright \Sigma_1[])$, for some positive or simple context $\Sigma_1[]$ and some structure W. The cases when $\Sigma[] = \circ(\Sigma_1[] \triangleright W)$ or $\Sigma[] = \circ(W \triangleright \Sigma_1[])$ are analogous, and the other cases are unchanged from the proof Lemma 4.1.4.

1. • Case when $\Sigma[] = \bullet(\Sigma_1[] \triangleright W)$. We first obtain the following derivation:

$$\cfrac{\cfrac{X \Rightarrow \bullet(\Sigma_1[p] \triangleright W)}{\circ(X \triangleright \emptyset), \Sigma_1[p] \Rightarrow W}\, rp_\bullet}{\Sigma_1[p] \Rightarrow (\circ(X \triangleright \emptyset) \triangleright W)}\, rp^{\triangleright}_{R}$$

If $\Sigma_1[]$ is a positive context, we apply statement (2) of the induction hypothesis to it; otherwise it must be a simple context and we apply Lemma 7.1.3. In both cases, we obtain the required derivation:

$$\begin{array}{c}\cfrac{\cfrac{X \Rightarrow \bullet(\Sigma_1[p] \triangleright W)}{\circ(X \triangleright \emptyset), \Sigma_1[p] \Rightarrow W}\, rp_\bullet}{\Sigma_1[p] \Rightarrow (\circ(X \triangleright \emptyset) \triangleright W)}\, rp^{\triangleright}_{R} \\ \vdots \\ Y \Rightarrow p\end{array}$$

• Case when $\Sigma[] = \bullet(W \triangleright \Sigma_1[])$. We first obtain the following derivation:

$$\cfrac{X \Rightarrow \bullet(W \triangleright \Sigma_1[p])}{\circ(X \triangleright \emptyset), W \Rightarrow \Sigma_1[p]}\, rp_\bullet$$

If $\Sigma_1[]$ is a positive context, we apply statement (1) of the induction hypothesis to it; otherwise it must be a simple context and we apply Lemma 7.1.3. In both cases, we obtain the required derivation:

$$\begin{array}{c}\cfrac{X \Rightarrow \bullet(W \triangleright \Sigma_1[p])}{\circ(X \triangleright \emptyset), W \Rightarrow \Sigma_1[p]}\, rp_\bullet \\ \vdots \\ Y \Rightarrow p\end{array}$$

2. • Case when $\Sigma[] = \bullet(\Sigma_1[] \triangleright W)$. We first obtain the following derivation:

$$\dfrac{\bullet(\Sigma_1[p] \triangleright W) \Rightarrow X}{\Sigma_1[p] \Rightarrow W, \circ(\emptyset \triangleright X)}\, rp_\circ$$

If $\Sigma_1[]$ is a positive context, we apply statement (2) of the induction hypothesis to it; otherwise it must be a simple context and we apply Lemma 7.1.3. In both cases, we obtain the required derivation:

$$\begin{array}{c}\dfrac{\bullet(\Sigma_1[p] \triangleright W) \Rightarrow X}{\Sigma_1[p] \Rightarrow W, \circ(\emptyset \triangleright X)}\, rp_\circ \\ \vdots \\ Z \Rightarrow p\end{array}$$

• Case when $\Sigma[] = \bullet(W \triangleright \Sigma_1[])$. We first obtain the following derivation:

$$\dfrac{\dfrac{\bullet(W \triangleright \Sigma_1[p]) \Rightarrow X}{W \Rightarrow \Sigma_1[p], \circ(\emptyset \triangleright X)}\, rp_\circ}{(W \triangleright \circ(\emptyset \triangleright X)) \Rightarrow \Sigma_1[p]}\, rp_L^\triangleright$$

If $\Sigma_1[]$ is a positive context, we apply statement (1) of the induction hypothesis to it; otherwise it must be a simple context and we apply Lemma 7.1.3. In both cases, we obtain the required derivation:

$$\begin{array}{c}\dfrac{\dfrac{\bullet(W \triangleright \Sigma_1[p]) \Rightarrow X}{W \Rightarrow \Sigma_1[p], \circ(\emptyset \triangleright X)}\, rp_\circ}{(W \triangleright \circ(\emptyset \triangleright X)) \Rightarrow \Sigma_1[p]}\, rp_L^\triangleright \\ \vdots \\ Z \Rightarrow p\end{array}$$

Q.E.D.

Lemma 7.1.5 (Display property for negative contexts)**.** *Let $\Sigma[]$ be a negative context. Let X be a structure and p a propositional variable not occurring in X nor $\Sigma[]$. Then there exist structures Y and Z such that:*

1. *$p \Rightarrow Y$ is DP-derivable from $X \Rightarrow \Sigma[p]$ and*
2. *$p \Rightarrow Z$ is DP-derivable from $\Sigma[p] \Rightarrow X$.*

Proof. Analogous to the proof of Lemma 7.1.4. Q.E.D.

Note that since the rules in DP are all invertible, the derivations constructed in the above lemmas are invertible derivations. That is, we can derive $Y \Rightarrow p$ from $X \Rightarrow \Sigma[p]$ and vice versa. Note also that since rules in the shallow system are closed under substitution, this also means $Y \Rightarrow W$ is derivable from $X \Rightarrow \Sigma[W]$, and vice versa, for any structure W.

7.2 Cut elimination in LBiKt

Our cut-elimination proof is based on the method of proof-substitution presented in Chapter 4; here we extend it to the cases involving the modal connectives $\Box$, $\Diamond$, $\blacksquare$, $\blacklozenge$ and $\circ$- and $\bullet$-structures. We illustrate one case with an example.

Consider the derivation below ending with a cut on $\Diamond A$:

$$\dfrac{\begin{array}{c}\Pi_1\\ X_1 \Rightarrow Y_1, \Diamond A\end{array} \qquad \begin{array}{c}\Pi_2\\ \Diamond A, X_2 \Rightarrow Y_2\end{array}}{X_1, X_2 \Rightarrow Y_1, Y_2}\; cut$$

Instead of permuting the cut rule locally, we trace the cut formula $\Diamond A$ until it becomes principal in the derivations Π_1 and Π_2, and then apply cut on a smaller formula. Suppose that Π_1 and Π_2 are respectively the two derivations below:

$$\begin{array}{c}\dfrac{\begin{array}{c}\Psi_1\\ (X_1' \rhd Y_1') \Rightarrow A\end{array}}{\circ(X_1' \rhd Y_1') \Rightarrow \Diamond A}\;\Diamond_R\\ \vdots \\ X_1 \Rightarrow Y_1, \Diamond A\end{array} \qquad \begin{array}{c}\dfrac{\begin{array}{c}\Psi_2\\ \circ(A \rhd \emptyset) \Rightarrow Y_2'\end{array}}{\Diamond A \Rightarrow Y_2'}\;\Diamond_L\\ \vdots \\ \Diamond A, X_2 \Rightarrow Y_2\end{array}$$

We first transform Π_1 by substituting $(X_2 \rhd Y_2)$ for $\Diamond A$ in Π_1 and obtain the sub-derivation below with an open leaf:

$$\begin{array}{c}\circ(X_1' \rhd Y_1') \Rightarrow (X_2 \rhd Y_2)\\ \vdots \\ \dfrac{X_1 \Rightarrow Y_1, (X_2 \rhd Y_2)}{X_1, X_2 \Rightarrow Y_1, Y_2}\; s_R\end{array}$$

We then prove the open leaf by uniformly substituting $\circ(X_1' \rhd Y_1')$ for $\Diamond A$ in Π_2, and applying cut on a sub-formula A:

$$\begin{array}{c}\dfrac{\dfrac{\dfrac{\begin{array}{c}\Psi_1\\ (X_1' \rhd Y_1') \Rightarrow A\end{array} \qquad \dfrac{\begin{array}{c}\Psi_2\\ \circ(A \rhd \emptyset) \Rightarrow Y_2'\end{array}}{A \Rightarrow \bullet(\emptyset \rhd Y_2')}\; rp_\bullet}{(X_1' \rhd Y_1') \Rightarrow \bullet(\emptyset \rhd Y_2')}\; cut}{X_1' \Rightarrow Y_1', \bullet(\emptyset \rhd Y_2')}\; s_L}{\circ(X_1' \rhd Y_1') \Rightarrow Y_2'}\; rp_\bullet\\ \vdots \\ \dfrac{\circ(X_1' \rhd Y_1'), X_2 \Rightarrow Y_2}{\circ(X_1' \rhd Y_1') \Rightarrow (X_2 \rhd Y_2)}\; \rhd_R\end{array}$$

The *cut rank* of an instance of cut is the size of the cut formula, as usual. The cut rank $cr(\Pi)$ of a derivation Π is the largest cut rank of the cut instances in Π (or zero, if Π is cut-free). Given a formula A, we denote with $|A|$ its size.

To formalise the cut elimination proof, we first introduce a notion of multiple-hole contexts. A k-hole context is a context with k holes. Given a k-hole context $Z[\cdots]$ we write $Z[X^k]$ to stand for the structure obtained from $Z[\cdots]$ by replacing each hole with an occurrence of the structure X.

Example 7.2.1. *If* $\Sigma[] = ([], \bullet([], W \triangleright Y)) \triangleright V$ *then* $\Sigma[(\circ(A \triangleright B))^2] = (\circ(A \triangleright B), \bullet(\circ(A \triangleright B), W \triangleright Y)) \triangleright V$.

A k-hole context is positive if every hole in it has positive polarity, and it is *quasi-positive* if every hole in it is either neutral or positive. A k-hole context is negative if every hole in it has negative polarity, and it is *quasi-negative* if every hole in it is either neutral or negative.

Example 7.2.2. *The 2-hole context* $([], \bullet([], W \triangleright Y)) \triangleright V$ *is quasi-negative. The 2-hole context* $(Y \triangleright \bullet[]) \triangleright (V \triangleright [])$ *is positive.*

Lemma 7.2.3 states the proof substitutions needed to eliminate atomic cuts. Lemmas 7.2.4-7.2.11 state the proof substitutions needed for non-atomic cuts. We only give the proofs of the cases involving the modal connectives as the other proofs are unchanged from Chapter 4.

Lemma 7.2.3. *Suppose* $p, X \Rightarrow Y$ *is cut-free derivable for some fixed* p, X *and* Y. *Then for any k-hole positive context* $Z_1[\cdots]$ *and any l-hole quasi-positive context* $Z_2[\cdots]$, *if* $Z_1[p^k] \Rightarrow Z_2[p^l]$ *is cut-free derivable, then* $Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ *is cut-free derivable.*

Proof. Analogous to the proof of Lemma 4.2.1. Q.E.D.

Lemma 7.2.4. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_i : X \Rightarrow Y, A_i$, *for some* $i \in \{1,2\}$, *such that* $cr(\Pi_i) < |A_1 \vee A_2|$. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_3 : Z_1[(A_1 \vee A_2)^k] \Rightarrow Z_2[(A_1 \vee A_2)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$, *such that* $cr(\Pi_3) < |A_1 \vee A_2|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ *and* $cr(\Pi) < |A \vee B|$.

Proof. Analogous to the proof of Lemma 4.2.2. Q.E.D.

Lemma 7.2.5. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : X \Rightarrow Y, A_1$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : X \Rightarrow Y, A_2$ *with* $cr(\Pi_1) < |A_1 \wedge A_2|$ *and* $cr(\Pi_2) < |A_1 \wedge A_2|$. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_3 : Z_1[(A_1 \wedge A_2)^k] \Rightarrow Z_2[(A_1 \wedge A_2)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$ *with* $cr(\Pi_3) < |A_1 \wedge A_2|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ *and* $cr(\Pi) < |A \wedge B|$.

Proof. Analogous to the proof of Lemma 4.2.3. Q.E.D.

Lemma 7.2.6. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : X, A \Rightarrow B$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : Z_1[(A \rightarrow B)^k] \Rightarrow Z_2[(A \rightarrow B)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|A \rightarrow B|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[X^k] \Rightarrow Z_2[X^l]$ *and* $cr(\Pi) < |A \rightarrow B|$.

Proof. Analogous to the proof of Lemma 4.2.4. Q.E.D.

Lemma 7.2.7. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : X \Rightarrow Y, A$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : X, B \Rightarrow Y$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|A \prec B|$. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_3 : Z_1[(A \prec B)^k] \Rightarrow Z_2[(A \prec B)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$ *with* $cr(\Pi_3) < |A \prec B|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ *and* $cr(\Pi) < |A \prec B|$.

Proof. Analogous to the proof of Lemma 4.2.5. Q.E.D.

Lemma 7.2.8. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : X \Rightarrow \circ(\emptyset \triangleright A)$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : Z_1[(\Box A)^k] \Rightarrow Z_2[(\Box A)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\Box A|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[X^k] \Rightarrow Z_2[X^l]$ *and* $cr(\Pi) < |\Box A|$.

Proof. By induction on $|\Pi_2|$. The non-trivial case is when Π_2 ends with $\Box_L$ on $\Box A$:

$$\dfrac{\begin{matrix}\Psi \\ A \Rightarrow Z_2'[(\Box A)^l]\end{matrix}}{\Box A \Rightarrow \circ(Z_2'[(\Box A)^l])}\ \Box_L$$

By induction hypothesis we have $\vdash_{\mathbf{LBiKt}} \Psi' : A \Rightarrow Z_2'[X^l]$ where $cr(\Psi') < |\Box A|$. The derivation Π is constructed as follows:

$$\dfrac{\dfrac{\dfrac{\begin{matrix}\Pi_1 \\ X \Rightarrow \circ(\emptyset \triangleright A)\end{matrix}}{\bullet(X \triangleright \emptyset) \Rightarrow A}\ rp_\circ \qquad \begin{matrix}\Psi' \\ A \Rightarrow Z_2'[X^l]\end{matrix}}{\bullet(X \triangleright \emptyset) \Rightarrow Z_2'[X^l]}\ cut}{X \Rightarrow \circ(Z_2'[X^l])}\ rp_\circ$$

Q.E.D.

Lemma 7.2.9. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : X \Rightarrow \bullet(\emptyset \triangleright A)$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : Z_1[(\blacksquare A)^k] \Rightarrow Z_2[(\blacksquare A)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\blacksquare A|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[X^k] \Rightarrow Z_2[X^l]$ *and* $cr(\Pi) < |\blacksquare A|$.

Proof. Analogous to the proof of Lemma 7.2.8. Q.E.D.

Lemma 7.2.10. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : (X_1 \triangleright X_2) \Rightarrow A$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : Z_1[(\Diamond A)^k] \Rightarrow Z_2[(\Diamond A)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$, *and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\Diamond A|$. *Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[(\circ(X_1 \triangleright X_2))^k] \Rightarrow Z_2[(\circ(X_1 \triangleright X_2))^l]$ *and* $cr(\Pi) < |\Diamond A|$.

Proof. By induction on $|\Pi_2|$. The non-trivial case is when Π_2 ends with $\Diamond_L$ on $\Diamond A$:

$$\dfrac{\begin{matrix}\Psi \\ \circ(A \triangleright \emptyset) \Rightarrow Z_2[(\Diamond A)^l]\end{matrix}}{\Diamond A \Rightarrow Z_2[(\Diamond A)^l]}\ \Diamond_L$$

By induction hypothesis we have $\vdash_{\mathbf{LBiKt}} \Psi' : \circ(A \triangleright \emptyset) \Rightarrow Z_2[(\circ(X_1 \triangleright X_2))^l]$ where $cr(\Psi') < |\Diamond A|$. The derivation Π is constructed as follows:

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi_1\\(X_1 \triangleright X_2) \Rightarrow A\end{array} \qquad \dfrac{\begin{array}{c}\Psi'\\ \circ(A \triangleright \emptyset) \Rightarrow Z_2[(\circ(X_1 \triangleright X_2))^l]\end{array}}{A \Rightarrow \bullet(\emptyset \triangleright Z_2[(\circ(X_1 \triangleright X_2))^l])}\,rp_\bullet}{(X_1 \triangleright X_2) \Rightarrow \bullet(\emptyset \triangleright Z_2[(\circ(X_1 \triangleright X_2))^l])}\,cut}{X_1 \Rightarrow X_2, \bullet(\emptyset \triangleright Z_2[(\circ(X_1 \triangleright X_2))^l])}\,s_L}{\circ(X_1 \triangleright X_2) \Rightarrow Z_2[(\circ(X_1 \triangleright X_2))^l]}\,rp_\bullet$$

Q.E.D.

Lemma 7.2.11. *Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : (X_1 \triangleright X_2) \Rightarrow A$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : Z_1[(\blacklozenge A)^k] \Rightarrow Z_2[(\blacklozenge A)^l]$ *for some k-hole quasi-negative context* $Z_1[\cdots]$ *and l-hole negative context* $Z_2[\cdots]$*, and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|\blacklozenge A|$*. Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[(\bullet(X_1 \triangleright X_2))^k] \Rightarrow Z_2[(\bullet(X_1 \triangleright X_2))^l]$ *and* $cr(\Pi) < |\blacklozenge A|$*.*

Proof. Analogous to the proof of Lemma 7.2.10. Q.E.D.

Lemma 7.2.12. *Let A be a non-atomic formula. Suppose* $\vdash_{\mathbf{LBiKt}} \Pi_1 : A, X \Rightarrow Y$ *and* $\vdash_{\mathbf{LBiKt}} \Pi_2 : Z_1[A^k] \Rightarrow Z_2[A^l]$ *where* $Z_1[\cdots]$ *is a k-hole positive context,* $Z_2[\cdots]$ *is an l-hole quasi-positive context, and the cut ranks of* Π_1 *and* Π_2 *are smaller than* $|A|$*. Then there exists* Π *such that* $\vdash_{\mathbf{LBiKt}} \Pi : Z_1[(X \triangleright Y)^k] \Rightarrow Z_2[(X \triangleright Y)^l]$ *and* $cr(\Pi) < |A|$*.*

Proof. By induction on $|\Pi_2|$ and case analysis on A. The non-trivial case is when Π_2 ends with a right-introduction rule on A. That is, in this case, we have $Z_2[A^l] = (Z_2'[A^{l-1}], A)$ for some positive context $Z_2'[\cdots]$. We distinguish several cases depending on A. We show here the cases where A is either $\Box C$ or $\Diamond C$.

- Suppose $A = \Box C$ and Π_2 is

  $$\dfrac{\begin{array}{c}\Psi\\ Z_1[(\Box C)^k] \Rightarrow \circ(\emptyset \triangleright C)\end{array}}{Z_1[(\Box C)^k] \Rightarrow \Box C}\,\Box_R$$

  By induction hypothesis, we have a derivation Ψ' of

  $$Z_1[(X \triangleright Y)^k] \Rightarrow \circ(\emptyset \triangleright C)$$

  Then the derivation Π is constructed as follows:

  $$\dfrac{\begin{array}{c}\theta\\ Z_1[(X \triangleright Y)^k], X \Rightarrow Y\end{array}}{Z_1[(X \triangleright Y)^k] \Rightarrow X \triangleright Y}\,\triangleright_R$$

  with θ obtained by applying Lemma 7.2.8 to Ψ' and Π_1.

- Suppose $A = \Diamond C$ and Π_2 is

  $$\dfrac{\begin{array}{c}\Psi\\ Z_1'[(\Diamond C)^k] \Rightarrow C\end{array}}{\circ(Z_1'[(\Diamond C)^k]) \Rightarrow \Diamond C}\ \Diamond_R$$

  By induction hypothesis, we have a derivation Ψ' of

  $$Z_1'[(X \triangleright Y)^k] \Rightarrow C$$

  Then the derivation Π is constructed as follows:

  $$\dfrac{\begin{array}{c}\theta\\ \circ(Z_1'[(X \triangleright Y)^k]), X \Rightarrow Y\end{array}}{\circ(Z_1'[(X \triangleright Y)^k]) \Rightarrow X \triangleright Y}\ \triangleright_R$$

  with θ obtained by applying Lemma 7.2.10 to Ψ' and Π_1.

The other cases are treated analogously, using Lemmas 7.2.4, 7.2.5, 7.2.6, 7.2.7, 7.2.9 and 7.2.11. Q.E.D.

Theorem 7.2.13. *If $X \Rightarrow Y$ is* **LBiKt***-derivable then it is also* **LBiKt***-derivable without using cut.*

Proof. As typical in cut elimination proofs, we remove topmost cuts in succession. Let Π be a derivation of **LBiKt** with a topmost cut instance

$$\dfrac{\begin{array}{c}\Pi_1\\ X_1 \Rightarrow Y_1, A\end{array} \qquad \begin{array}{c}\Pi_2\\ A, X_2 \Rightarrow Y_2\end{array}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ cut$$

Note that Π_1 and Π_2 are both cut-free since this is a topmost instance in Π. We use induction on the size of A to eliminate this topmost instance of cut.

If A is an atomic formula p then the cut free derivation is constructed as follows where Ψ is obtained from applying Lemma 7.2.3 to Π_2 and Π_1:

$$\dfrac{\begin{array}{c}\Psi\\ X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)\end{array}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R$$

If A is non-atomic, using Lemma 7.2.12 we get the following derivation Π':

$$\dfrac{\begin{array}{c}\Psi\\ X_1 \Rightarrow Y_1, (X_2 \triangleright Y_2)\end{array}}{X_1, X_2 \Rightarrow Y_1, Y_2}\ s_R$$

We have $cr(\Pi') < |A|$ by Lemma 7.2.12, therefore by induction hypothesis, we can remove all the cuts in Π' to get a cut-free derivation of $X_1, X_2 \Rightarrow Y_1, Y_2$. Q.E.D.

7.3 Equivalence of DBiKt and LBiKt

In this section we show the equivalence of **DBiKt** and **LBiKt**.

7.3.1 Soundness of DBiKt

We show that every derivation in **DBiKt** can be mimicked by a cut-free derivation in **LBiKt**. The non-trivial cases involve showing that the propagation rules of **DBiKt** are derivable in **LBiKt** using residuation. This is not surprising since the residuation rules in display calculi are used exactly for the purpose of displaying and un-displaying sub-structures so that inference rules can be applied to them.

Theorem 7.3.1 (Soundness)**.** *For any structures X and Y, if* $\vdash_{\mathbf{DBiKt}} \Pi : X \triangleright Y$ *then* $\vdash_{\mathbf{LBiKt}} \Pi' : X \Rightarrow Y$.

Proof. We show that each deep inference rule ρ is derivable in the shallow system. This is done by case analysis of the context $\Sigma[\,]$ in which the deep rule ρ applies. Note that if a deep inference rule ρ is applicable to $X \triangleright Y$, then the context $\Sigma[\,]$ in this case is either $[\,]$, a positive context or a negative context.

- For the case where $\Sigma[\,]$ is either positive or negative, we use the display property. We show here the case where ρ is a rule with a single premise; the other cases are analogous. Suppose ρ is

  $$\frac{\Sigma^+[U]}{\Sigma^+[V]}\,\rho$$

  By the display properties, we need only to show that the following rules are derivable in the shallow system for some structure W':

  $$\frac{W' \Rightarrow U}{W' \Rightarrow V} \qquad \frac{U \Rightarrow W'}{V \Rightarrow W'}$$

  For example, to show soundness of $\Box_{L2}$ it is enough to show that the following are derivable:

  $$\frac{W' \Rightarrow (\Box A, X \triangleright \circ(A, Y \triangleright Z), W)}{W' \Rightarrow (\Box A, X \triangleright \circ(Y \triangleright Z), W)} \qquad \frac{((\Box A, X \triangleright \circ(A, Y \triangleright Z), W) \Rightarrow W'}{(\Box A, X \triangleright \circ(Y \triangleright Z), W) \Rightarrow W'}$$

  Both reduce to showing that the following is derivable:

  $$\frac{\Box A, X \triangleright \circ(A, Y \triangleright Z), W}{\Box A, X \triangleright \circ(Y \triangleright Z), W}$$

- For the case where $\Sigma[\,] = []$, we only need to show that each valid instance of ρ where $\Sigma[\,] = [\,]$ is derivable in the shallow system **LBiKt**.

We now give derivations for all the non-trivial cases, including $\Box_{L2}$.

- The cases for propagation rules $\triangleright_{L1}$, $\triangleright_{R1}$, $\triangleright_{L2}$, $\triangleright_{R2}$ remain unchanged from the proof of Theorem 5.2.1.

- Rule $\Box_{L1}$:

$$\dfrac{Z, A, \bullet(\Box A, X \triangleright Y) \triangleright W}{Z, \bullet(\Box A, X \triangleright Y) \triangleright W}\,\Box_{L1} \quad \rightsquigarrow \quad \begin{array}{cl}
Z, A, \bullet(\Box A, X \triangleright Y) \Rightarrow W & \\ \hline
A, \bullet(\Box A, X \triangleright Y) \Rightarrow (Z \triangleright W) & \triangleright_R \\ \hline
A \Rightarrow (\bullet(\Box A, X \triangleright Y) \triangleright (Z \triangleright W)) & \triangleright_R \\ \hline
\Box A \Rightarrow \circ(\bullet(\Box A, X \triangleright Y) \triangleright (Z \triangleright W)) & \Box_L \\ \hline
\Box A \Rightarrow \circ(\bullet(\Box A, X \triangleright Y) \triangleright (Z \triangleright W)), Y & w_R \\ \hline
\Box A, X \Rightarrow \circ(\bullet(\Box A, X \triangleright Y) \triangleright (Z \triangleright W)), Y & w_L \\ \hline
(\Box A, X \triangleright Y) \Rightarrow \circ(\bullet(\Box A, X \triangleright Y) \triangleright (Z \triangleright W)) & \triangleright_L \\ \hline
\bullet(\Box A, X \triangleright Y) \Rightarrow (\bullet(\Box A, X \triangleright Y) \triangleright (Z \triangleright W)) & rp'_\circ \\ \hline
\bullet(\Box A, X \triangleright Y), \bullet(\Box A, X \triangleright Y) \Rightarrow (Z \triangleright W) & s_R \\ \hline
\bullet(\Box A, X \triangleright Y) \Rightarrow (Z \triangleright W) & c_L \\ \hline
Z, \bullet(\Box A, X \triangleright Y) \Rightarrow W & s_R
\end{array}$$

- Rule $\blacksquare_{L1}$: analogous to the case for $\Box_{L1}$

- Rule $\Diamond_{R1}$:

$$\dfrac{Z \triangleright \bullet(Y \triangleright \Diamond A, X), A, W}{Z \triangleright \bullet(Y \triangleright \Diamond A, X), W}\,\Diamond_{R1} \quad \rightsquigarrow \quad \begin{array}{cl}
Z \Rightarrow \bullet(Y \Rightarrow \Diamond A, X), A, W & \\ \hline
(Z \triangleright W) \Rightarrow \bullet(Y \triangleright \Diamond A, X), A & \triangleright_L \\ \hline
((Z \triangleright W) \triangleright \bullet(Y \triangleright \Diamond A, X)) \Rightarrow A & \triangleright_L \\ \hline
\circ((Z \triangleright W) \triangleright \bullet(Y \triangleright \Diamond A, X)) \Rightarrow \Diamond A & \Diamond_R \\ \hline
\circ((Z \triangleright W) \triangleright \bullet(Y \triangleright \Diamond A, X)) \Rightarrow \Diamond A, X & w_R \\ \hline
Y, \circ((Z \triangleright W) \triangleright \bullet(Y \triangleright \Diamond A, X)) \Rightarrow \Diamond A, X & w_L \\ \hline
\circ((Z \triangleright W) \triangleright \bullet(Y \triangleright \Diamond A, X)) \Rightarrow (Y \triangleright \Diamond A, X) & \triangleright_R \\ \hline
((Z \triangleright W) \triangleright \bullet(Y \triangleright \Diamond A, X)) \Rightarrow \bullet(Y \triangleright \Diamond A, X) & rp'_\bullet \\ \hline
(Z \triangleright W) \Rightarrow \bullet(Y \triangleright \Diamond A, X), \bullet(Y \triangleright \Diamond A, X) & s_L \\ \hline
(Z \triangleright W) \Rightarrow \bullet(Y \triangleright \Diamond A, X) & c_R \\ \hline
Z \Rightarrow \bullet(Y \triangleright \Diamond A, X), W & s_L
\end{array}$$

- Rule $\blacklozenge_{R1}$: analogous to the case for $\Diamond_{R1}$

- Rule $\Box_{L2}$:

$$\dfrac{\Box A, X \rhd \circ(A, Y \rhd Z), W}{\Box A, X \rhd \circ(Y \rhd Z), W}\,\Box_{L2} \quad \rightsquigarrow \quad \begin{array}{cl}
\Box A, X \Rightarrow \circ(A, Y \rhd Z), W & \\ \hline
A, Y, \bullet(\Box A, X \rhd W) \Rightarrow Z & rp_{\circ} \\ \hline
A, \bullet(\Box A, X \rhd W) \Rightarrow (Y \rhd Z) & \rhd_R \\ \hline
A \Rightarrow \bullet(\Box A, X \rhd W) \rhd (Y \rhd Z) & \rhd_R \\ \hline
\Box A \Rightarrow \circ(\bullet(\Box A, X \rhd W) \rhd (Y \rhd Z)) & \Box_L \\ \hline
\Box A, X \Rightarrow \circ(\bullet(\Box A, X \rhd W) \rhd (Y \rhd Z)) & w_L \\ \hline
\Box A, X \Rightarrow \circ(\bullet(\Box A, X \rhd W) \rhd (Y \rhd Z)), W & w_R \\ \hline
(\Box A, X \rhd W) \Rightarrow \circ(\bullet(\Box A, X \rhd W) \rhd (Y \rhd Z)) & \rhd_L \\ \hline
\bullet(\Box A, X \rhd W) \Rightarrow \bullet(\Box A, X \rhd W) \rhd (Y \rhd Z) & rp'_{\circ} \\ \hline
\bullet(\Box A, X \rhd W), \bullet(\Box A, X \rhd W) \Rightarrow (Y \rhd Z) & s_R \\ \hline
\bullet(\Box A, X \rhd W) \Rightarrow (Y \rhd Z) & c_L \\ \hline
(\Box A, X \rhd W) \Rightarrow \circ(Y \rhd Z) & rp'_{\circ} \\ \hline
\Box A, X \Rightarrow \circ(Y \rhd Z), W & s_L
\end{array}$$

- Rule $\blacksquare_{L2}$: analogous to the case for $\Box_{L2}$

- Rule $\Diamond_{R2}$:

$$\dfrac{X, \circ(Y \rhd Z, A) \rhd W, \Diamond A}{X, \circ(Y \rhd Z) \rhd W, \Diamond A}\,\Diamond_{R2} \quad \rightsquigarrow \quad \begin{array}{cl}
X, \circ(Y \rhd Z, A) \Rightarrow W, \Diamond A & \\ \hline
Y \Rightarrow \bullet(X \rhd W, \Diamond A), Z, A & rp_{\bullet} \\ \hline
(Y \rhd Z) \Rightarrow \bullet(X \rhd W, \Diamond A), A & \rhd_L \\ \hline
(Y \rhd Z) \rhd \bullet(X \rhd W, \Diamond A) \Rightarrow A & \rhd_L \\ \hline
\circ((Y \rhd Z) \rhd \bullet(X \rhd W, \Diamond A)) \Rightarrow \Diamond A & \Diamond_R \\ \hline
\circ((Y \rhd Z) \rhd \bullet(X \rhd W, \Diamond A)) \Rightarrow W, \Diamond A & w_R \\ \hline
X, \circ((Y \rhd Z) \rhd \bullet(X \rhd W, \Diamond A)) \Rightarrow W, \Diamond A & w_L \\ \hline
\circ((Y \rhd Z) \rhd \bullet(X \rhd W, \Diamond A)) \Rightarrow (X \rhd W, \Diamond A) & \rhd_R \\ \hline
((Y \rhd Z) \rhd \bullet(X \rhd W, \Diamond A)) \Rightarrow \bullet(X \rhd W, \Diamond A) & rp'_{\bullet} \\ \hline
(Y \rhd Z) \Rightarrow \bullet(X \rhd W, \Diamond A), \bullet(X \rhd W, \Diamond A) & s_L \\ \hline
(Y \rhd Z) \Rightarrow \bullet(X \rhd W, \Diamond A) & c_R \\ \hline
\circ(Y \rhd Z) \Rightarrow (X \rhd W, \Diamond A) & rp'_{\bullet} \\ \hline
X, \circ(Y \rhd Z) \Rightarrow W, \Diamond A & s_R
\end{array}$$

- Rule $\blacklozenge_{R2}$: analogous to the case for $\Diamond_{R2}$

- Rule $-\!\!<_L$:

$$\dfrac{Z, A-\!\!<B, (A \rhd B) \rhd Y}{Z, A-\!\!<B \rhd Y}\,-\!\!<_L \quad\rightsquigarrow\quad \cfrac{\cfrac{\cfrac{\cfrac{\cfrac{Z, A-\!\!<B, (A \rhd B) \Rightarrow Y}{(A \rhd B) \Rightarrow (Z, A-\!\!<B \rhd Y)}\,\rhd_R}{A \Rightarrow B, (Z, A-\!\!<B \rhd Y)}\,s_L}{A-\!\!<B \Rightarrow (Z, A-\!\!<B \rhd Y)}\,-\!\!<_L}{Z, A-\!\!<B, A-\!\!<B \Rightarrow Y}\,s_R}{Z, A-\!\!<B \Rightarrow Y}\,c_L$$

- Rule $\to_R$: analogous to the case for $-\!\!<_L$
- Rule $-\!\!<_R$: analogous to the case for $\to_L$
- Rule $\Diamond_L$:

$$\dfrac{Z, \Diamond A, \circ(A \rhd \emptyset) \rhd Y}{Z, \Diamond A \rhd Y}\,\Diamond_L \quad\rightsquigarrow\quad \cfrac{\cfrac{\cfrac{\cfrac{Z, \Diamond A, \circ(A \rhd \emptyset) \Rightarrow Y}{\circ(A \rhd \emptyset) \Rightarrow (Z, \Diamond A \rhd Y)}\,\rhd_R}{\Diamond A \Rightarrow (Z, \Diamond A \rhd Y)}\,\Diamond_L}{Z, \Diamond A, \Diamond A \Rightarrow Y}\,s_R}{Z, \Diamond A \Rightarrow Y}\,c_L$$

- Rules $\Box_R$, $\blacklozenge_L$, $\blacksquare_R$: analogous the case for $\Diamond_L$

Q.E.D.

7.3.2 Completeness of DBiKt

We now show that any *cut-free* **LBiKt**-derivation can be transformed into a cut-free **DBiKt**-derivation. This requires proving cut-free admissibility of various structural rules in **DBiKt**. The admissibility of general weakening and *formula* contraction (but not general contraction, which we will show later) is straightforward by induction on the height of derivations, just as for the calculus **DBiInt** in Chapter 5.

Lemma 7.3.2 (Admissibility of general weakening)**.** *For any structures X and Y: if $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[X]$ and $\Sigma[X, Y]$ is a structure, then $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[X, Y]$ such that $|\Pi'| = |\Pi|$.*

Proof. Analogous to the proof of Lemma 5.2.2. Q.E.D.

Lemma 7.3.3 (Admissibility of formula contraction)**.** *For any structure X and formula A: if $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[X, A, A]$ then $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[X, A]$ such that $|\Pi'| = |\Pi|$.*

Proof. Analogous to the proof of Lemma 5.2.3. Q.E.D.

Just as for the calculus **DBiInt** in Chapter 5, invertibility of the **DBiKt** rules follows immediately, since for each of our rules, the premise is a superset of the conclusion, and weakening is height-preserving.

Lemma 7.3.4 (Invertibility)**.** *All* **DBiKt** *rules are invertible: if the conclusion is derivable, then each premise is derivable.*

We now show that the residuation rules of **LBiKt** for $\triangleright$-structures are admissible in **DBiKt**: *i.e.* they can be simulated by the propagation rules of **DBiKt**. Actually, what we show next is admissibility of the "deep" versions of the residuation rules for $\triangleright$, which is important for showing the completeness of our proof search procedure that we introduce later.

Lemmas 7.3.5 to 7.3.8 are proved by structural induction on $\Sigma[]$, and a sub-induction on $|\Pi|$. We label a dashed line with the lemma used to obtain the conclusion from the premise, and obtain Π_1' from Π_1 using the sub-induction hypothesis.

Lemma 7.3.5 (Deep admissibility of s_L). *If* $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[(X \triangleright Y), Z \triangleright W]$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[X, Z \triangleright Y, W]$ *such that* $|\Pi'| \leq |\Pi|$.

Proof.
Only the base case when $\Sigma[] = []$ is non-trivial:

- Case when Π ends with a propagation rule $\blacksquare_{L2}$ that moves a formula between X and Y:

$$\dfrac{\begin{array}{c}\Pi_1\\ (\blacksquare A, X_1 \triangleright \bullet(A, Y_1 \triangleright Y_2)), Z \triangleright W\end{array}}{(\blacksquare A, X_1 \triangleright \bullet(Y_1 \triangleright Y_2)), Z \triangleright W}\ \blacksquare_{L2} \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \blacksquare A, X_1, Z \triangleright \bullet(A, Y_1 \triangleright Y_2), W\end{array}}{\blacksquare A, X_1, Z \triangleright \bullet(Y_1 \triangleright Y_2), W}\ \blacksquare_{L2}$$

- Case when Π ends with a propagation rule $\Box_{L2}$ that moves a formula between X and Y is analogous to the case for $\blacksquare_{L2}$.

- Case when Π ends with a propagation rule $\Diamond_{R2}$ that moves a formula between X and Y:

$$\dfrac{\begin{array}{c}\Pi_1\\ (\circ(X_1 \triangleright X_2, A) \triangleright Y_1, \Diamond A), Z \triangleright W\end{array}}{(\circ(X_1 \triangleright X_2) \triangleright Y_1, \Diamond A), Z \triangleright W}\ \Diamond_{R2} \quad \rightsquigarrow \quad \dfrac{\begin{array}{c}\Pi_1'\\ \circ(X_1 \triangleright X_2, A), Z \triangleright W, Y_1, \Diamond A\end{array}}{\circ(X_1 \triangleright X_2), Z \triangleright W, Y_1, \Diamond A}\ \Diamond_{R2}$$

- Case when Π ends with a propagation rule $\blacklozenge_{R2}$ that moves a formula between X and Y is analogous to the case for $\Diamond_{L2}$.

- Cases involving the propagation rules $\triangleright_{L1}, \triangleright_{R1}, \triangleright_{L2}, \triangleright_{R2}$ are unchanged from the proof of Lemma 5.2.5.

- Cases involving other propagation and logical rules follow immediately from the sub-induction hypothesis, since they do not move formulae across $\triangleright$-structures.

Q.E.D.

Lemma 7.3.6 (Deep admissibility of s_R). *If* $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[X \triangleright Y, (Z \triangleright W)]$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[X, Z \triangleright Y, W]$ *such that* $|\Pi'| \leq |\Pi|$.

Proof. Analogous to the proof of Lemma 7.3.5. Q.E.D.

Lemma 7.3.7 (Deep admissibility of $\triangleright_L$). *If* $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[X \triangleright Y, Z]$ *and* Σ *is either the empty context* $[\,]$ *or a negative context* $\Sigma_1^-[\,]$*, then* $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[(X \triangleright Y) \triangleright Z]$.

Proof.

- Case when $\Sigma[] = []$.
  - Case when Π ends with a propagation rule $\blacksquare_{L2}$ that moves a formula between X and Z:

$$\dfrac{\begin{array}{c}\Pi_1\\ \blacksquare A, X_1 \triangleright Y, \bullet(A, Z_1 \triangleright Z_2), Z_3\end{array}}{\blacksquare A, X_1 \triangleright Y, \bullet(Z_1 \triangleright Z_2), Z_3}\,\blacksquare_{L2} \qquad \rightsquigarrow$$

$$\dfrac{\dfrac{\begin{array}{c}\Pi'_1\\ (\blacksquare A, X_1 \triangleright Y) \triangleright \bullet(A, Z_1 \triangleright Z_2), Z_3\\ \hdashline \blacksquare A, (\blacksquare A, X_1 \triangleright Y) \triangleright \bullet(A, Z_1 \triangleright Z_2), Z_3\end{array}\ \text{Lemma 7.3.2}}{\blacksquare A, (\blacksquare A, X_1 \triangleright Y) \triangleright \bullet(Z_1 \triangleright Z_2), Z_3}\,\blacksquare_{L2}}{(\blacksquare A, X_1 \triangleright Y) \triangleright \bullet(Z_1 \triangleright Z_2), Z_3}\,\triangleright_{L1}$$

  - Case when Π ends with a propagation rule $\square_{L2}$ that moves a formula between X and Z is analogous to the case for $\blacksquare_{L2}$.
  - Case when Π ends with a propagation rule $\Diamond_{R2}$ that moves a formula between X and Z:

$$\dfrac{\begin{array}{c}\Pi_1\\ \circ(X_1 \triangleright X_2, A), X_3 \triangleright Y, Z_1, \Diamond A\end{array}}{\circ(X_1 \triangleright X_2), X_3 \triangleright Y, Z_1, \Diamond A}\,\Diamond_{R2} \qquad \rightsquigarrow$$

$$\dfrac{\dfrac{\begin{array}{c}\Pi'_1\\ (\circ(X_1 \triangleright X_2, A), X_3 \triangleright Y) \triangleright Z_1, \Diamond A\\ \hdashline (\circ(X_1 \triangleright X_2, A), X_3 \triangleright Y, \Diamond A) \triangleright Z_1, \Diamond A\end{array}\ \text{Lemma 7.3.2}}{(\circ(X_1 \triangleright X_2), X_3 \triangleright Y, \Diamond A) \triangleright Z_1, \Diamond A}\,\Diamond_{R2}}{(\circ(X_1 \triangleright X_2), X_3 \triangleright Y) \triangleright Z_1, \Diamond A}\,\triangleright_{R2}$$

  - Case when Π ends with a propagation rule $\blacklozenge_{R2}$ that moves a formula between X and Z is analogous to the case for $\Diamond_{R2}$.
  - Cases involving the propagation rules $\triangleright_{L1}$, $\triangleright_{R1}$, $\triangleright_{L2}$, $\triangleright_{R2}$ are unchanged from the proof of Lemma 5.2.7.
- Case when $\Sigma[] = \Sigma_1[U, [] \triangleright V]$ for some Σ_1.
  - Cases involving the propagation rules $\triangleright_{L1}$, $\triangleright_{R1}$, $\triangleright_{L2}$, $\triangleright_{R2}$ are unchanged from the proof of Lemma 5.2.7.

- Case when $\Sigma[] = \Sigma_1[U, \circ[] \rhd V]$ for some Σ_1.

  - Case when Π ends with a propagation rule $\Diamond_{R2}$ that moves a formula from outside the context into the context:

$$\dfrac{\begin{array}{c}\Pi_1\\ U, \circ(X \rhd Y, Z, A) \rhd V_1, \Diamond A\end{array}}{U, \circ(X \rhd Y, Z) \rhd V_1, \Diamond A}\,\Diamond_{R2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ U, \circ((X \rhd Y) \rhd Z, A) \rhd V_1, \Diamond A\end{array}}{U, \circ((X \rhd Y) \rhd Z) \rhd V_1, \Diamond A}\,\Diamond_{R2}$$

  - Case when Π ends with a propagation rule $\blacksquare_{L1}$ that moves a formula out of the context:

$$\dfrac{\begin{array}{c}\Pi_1\\ U, A, \circ(\blacksquare A, X_1 \rhd Y, Z) \rhd V\end{array}}{U, \circ(\blacksquare A, X_1 \rhd Y, Z) \rhd V}\,\blacksquare_{L1} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\Pi_1'\\ U, A, \circ((\blacksquare A, X_1 \rhd Y) \rhd Z) \rhd V\end{array}}{U, A, \circ(\blacksquare A, (\blacksquare A, X_1 \rhd Y) \rhd Z) \rhd V}\,\text{Lemma 7.3.2}}{U, \circ(\blacksquare A, (\blacksquare A, X_1 \rhd Y) \rhd Z) \rhd V}\,\blacksquare_{L1}}{U, \circ((\blacksquare A, X_1 \rhd Y) \rhd Z) \rhd V}\,\rhd_{L1}$$

- Case when $\Sigma[] = \Sigma_1[U, \bullet[] \rhd V]$ for some Σ_1.

  - Case when Π ends with a propagation rule $\blacklozenge_{R2}$ that moves a formula from outside the context into the context:

$$\dfrac{\begin{array}{c}\Pi_1\\ U, \bullet(X \rhd Y, Z, A) \rhd V_1, \blacklozenge A\end{array}}{U, \bullet(X \rhd Y, Z) \rhd V_1, \blacklozenge A}\,\blacklozenge_{R2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ U, \bullet((X \rhd Y) \rhd Z, A) \rhd V_1, \blacklozenge A\end{array}}{U, \bullet((X \rhd Y) \rhd Z) \rhd V_1, \blacklozenge A}\,\blacklozenge_{R2}$$

  - Case when Π ends with a propagation rule $\Box_{L1}$ that moves a formula out of the context:

$$\dfrac{\begin{matrix}\Pi_1\\ U, A, \bullet(\Box A, X_1 \rhd Y, Z) \rhd V\end{matrix}}{U, \bullet(\Box A, X_1 \rhd Y, Z) \rhd V}\,\Box_{L1} \qquad \leadsto$$

$$\dfrac{\dfrac{\begin{matrix}\Pi_1'\\ U, A, \bullet((\Box A, X_1 \rhd Y) \rhd Z) \rhd V\end{matrix}}{U, A, \bullet(\Box A, (\Box A, X_1 \rhd Y) \rhd Z) \rhd V}\,\text{Lemma 7.3.2}}{\dfrac{U, \bullet(\Box A, (\Box A, X_1 \rhd Y) \rhd Z) \rhd V}{U, \bullet((\Box A, X_1 \rhd Y) \rhd Z) \rhd V}\,\rhd_{L1}}\,\Box_{L1}$$

Q.E.D.

Lemma 7.3.8 (Deep admissibility of $\rhd_R$). *If* $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[X, Y \rhd Z]$ *and* Σ *is either the empty context* $[\,]$ *or a positive context* $\Sigma_1^+[\,]$*, then* $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[X \rhd (Y \rhd Z)]$.

Proof. Analogous to the proof of Lemma 7.3.7. Q.E.D.

We now show that the residuation rules of **LBiKt** for $\circ$- and $\bullet$-structures are admissible in **DBiKt**; that is, they can be simulated by the propagation rules of **DBiKt**. Lemmas 7.3.9 to 7.3.12 are proved by induction on $|\Pi|$, and Π_1' (Π_2' resp.) is obtained from Π_1 (Π_2 resp.) using the induction hypothesis.

Lemma 7.3.9 (Admissibility of $rp_\bullet$). *If* $\vdash_{\mathbf{DBiKt}} \Pi : X \rhd Y, \bullet(W \rhd Z)$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : \circ(X \rhd Y), W \rhd Z$ *such that* $|\Pi'| = |\Pi|$.

Proof.

- Case when Π ends with a propagation rule $\blacksquare_{L2}$ that moves a formula between X and $\bullet(W \rhd Z)$:

$$\dfrac{\begin{matrix}\Pi_1\\ X_1, \blacksquare A \rhd Y, \bullet(A, W \rhd Z)\end{matrix}}{X_1, \blacksquare A \rhd Y, \bullet(W \rhd Z)}\,\blacksquare_{L2} \quad \leadsto \quad \dfrac{\begin{matrix}\Pi_1'\\ \circ(X_1, \blacksquare A \rhd Y), A, W \rhd Z\end{matrix}}{\circ(X_1, \blacksquare A \rhd Y), W \rhd Z}\,\blacksquare_{L1}$$

- Case when Π ends with a propagation rule $\Diamond_{R1}$ that moves a formula out of $\bullet(W \rhd Z)$:

$$\dfrac{\begin{matrix}\Pi_1\\ X \rhd Y, A, \bullet(W \rhd Z_1, \Diamond A)\end{matrix}}{X \rhd Y, \bullet(W \rhd Z_1, \Diamond A)}\,\Diamond_{R1} \quad \leadsto \quad \dfrac{\begin{matrix}\Pi_1'\\ \circ(X \rhd Y, A), W \rhd Z_1, \Diamond A\end{matrix}}{\circ(X \rhd Y), W \rhd Z_1, \Diamond A}\,\Diamond_{R2}$$

- Case when Π ends with $\to_L$ rule whose principal formula is in X:

$$\dfrac{\overset{\Pi_1}{(X_1, A \to B) \triangleright A, Y, \bullet(W \triangleright Z)} \qquad \overset{\Pi_2}{X_1, A \to B, B \triangleright Y, \bullet(W \triangleright Z)}}{X_1, A \to B \triangleright Y, \bullet(W \triangleright Z)} \to_L \quad \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{\circ(X_1, A \to B \triangleright A, Y), W \triangleright Z} \qquad \overset{\Pi_2'}{\circ(X_1, A \to B, B \triangleright Y), W \triangleright Z}}{\circ(X_1, A \to B \triangleright Y), W \triangleright Z} \to_L$$

- Case when Π ends with $-\!\!<_R$ rule whose principal formula is in Z:

$$\dfrac{\overset{\Pi_1}{X \triangleright Y, \bullet(W \triangleright A, Z_1, A -\!\!< B)} \qquad \overset{\Pi_2}{X \triangleright Y, \bullet(B, W \triangleright Z_1, A -\!\!< B)}}{X \triangleright Y, \bullet(W \triangleright Z_1, A -\!\!< B)} -\!\!<_R \quad \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{\circ(X \triangleright Y), W \triangleright A, Z_1, A -\!\!< B} \qquad \overset{\Pi_2'}{\circ(X \triangleright Y), B, W \triangleright Z_1, A -\!\!< B}}{\circ(X \triangleright Y), W \triangleright Z_1, A -\!\!< B} -\!\!<_R$$

The cases involving other propagation and logical rules follow immediately from the induction hypothesis, since they do not move formulae into or out of $\bullet Z$. Q.E.D.

Lemma 7.3.10 (Admissibility of $rp_\bullet$). *If* $\vdash_{\mathbf{DBiKt}} \Pi : \circ(X \triangleright Y), W \triangleright Z$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : X \triangleright Y, \bullet(W \triangleright Z)$ *such that* $|\Pi'| = |\Pi|$.

Proof. Similar to the proof of Lemma 7.3.9. Q.E.D.

Lemma 7.3.11 (Admissibility of $rp_\circ$). *If* $\vdash_{\mathbf{DBiKt}} \Pi : X \triangleright Y, \circ(W \triangleright Z)$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : \bullet(X \triangleright Y), W \triangleright Z$ *such that* $|\Pi'| = |\Pi|$.

Proof. Analogous to the proof of Lemma 7.3.9. Q.E.D.

Lemma 7.3.12 (Admissibility of $rp_\circ$). *If* $\vdash_{\mathbf{DBiKt}} \Pi : \bullet(X \triangleright Y), W \triangleright Z$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : X \triangleright Y, \circ(W \triangleright Z)$ *such that* $|\Pi'| = |\Pi|$.

Proof. Analogous to the proof of Lemma 7.3.10. Q.E.D.

7.3.3 Admissibility of general contraction

The admissibility of general contraction on structures states that for any structure Y, if $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[Y, Y]$ then $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[Y]$. The proof of admissibility of general contraction again requires proving several distribution properties among structural connectives, similar to **DBiInt** in Chapter 5 and **DKt** in Chapter 6, but involving both the $\triangleright$ structural connective and either $\bullet$ or $\circ$. First, we need a basic distribution lemma for $\triangleright$-structures:

Lemma 7.3.13. *For any context* Σ *and any structures* X, Y, Z, W: *if* $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[(X \triangleright Y), (Z \triangleright W)]$ *then* $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[(X, Z \triangleright Y, W)]$ *such that* $|\Pi'| = |\Pi|$.

Proof. Analogous to the proof of Lemma 5.2.9. Q.E.D.

Now we prove two distribution lemmas that involve both tense and bi-intuitionistic structural connectives:

Lemma 7.3.14. *For any context Σ and any structures X, Y, Z, W: if $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[\circ(X \rhd Y), \circ(Z \rhd W)]$ then $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[\circ(X, Z \rhd Y, W)]$ such that $|\Pi'| = |\Pi|$.*

Proof. By induction on the height of Π. Similarly to the proof of Lemma 5.2.9, the non-trivial cases are when Π ends with a propagation rule applied to the structures X, Y, Z, W. We consider the cases where Σ is a positive context; the cases where Σ is a negative context are analogous.

- Case when Π ends with $\blacklozenge_{R1}$ rule that propagates a formula out of Y, note that we apply the induction hypothesis to the context $\Sigma^+[A, []]$:

$$\dfrac{\begin{matrix}\Pi_1\\ \Sigma^+[A, \circ(X \rhd Y_1, \blacklozenge A), \circ(Z \rhd W)]\end{matrix}}{\Sigma^+[\circ(X \rhd Y_1, \blacklozenge A), \circ(Z \rhd W)]}\,\blacklozenge_{R1} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi_1'\\ \Sigma^+[A, \circ(X, Z \rhd Y_1, \blacklozenge A, W)]\end{matrix}}{\Sigma^+[\circ(X, Z \rhd Y_1, \blacklozenge A, W)]}\,\blacklozenge_{R1}$$

- Case when Π ends with $\Box_{L2}$ rule that propagates a formula into $X \rhd Y$:

$$\dfrac{\begin{matrix}\Pi_1\\ \Sigma[U, \Box A \rhd \circ(A, X \rhd Y), \circ(Z \rhd W), V]\end{matrix}}{\Sigma[U, \Box A \rhd \circ(X \rhd Y), \circ(Z \rhd W), V]}\,\Box_{L2} \quad \rightsquigarrow$$

$$\dfrac{\begin{matrix}\Pi_1'\\ \Sigma[U, \Box A \rhd \circ(A, X, Z \rhd Y, W), V]\end{matrix}}{\Sigma[U, \Box A \rhd \circ(X, Z \rhd Y, W), V]}\,\Box_{L2}$$

Q.E.D.

Lemma 7.3.15. *For any context Σ and any structures X, Y, Z, W: if $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[\bullet(X \rhd Y), \bullet(Z \rhd W)]$ then $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[\bullet(X, Z \rhd Y, W)]$ such that $|\Pi'| = |\Pi|$.*

Proof. Analogous to the proof of Lemma 7.3.14. Q.E.D.

Finally, we can prove the main lemma of contraction admissibility. Some of the non-trivial cases use the distribution lemmas we just showed.

Lemma 7.3.16 (Admissibility of general contraction)**.** *For any structure Y: if $\vdash_{\mathbf{DBiKt}} \Pi : \Sigma[Y, Y]$ then $\vdash_{\mathbf{DBiKt}} \Pi' : \Sigma[Y]$ such that $|\Pi'| = |\Pi|$.*

Proof. By induction on the size of Y, with a sub-induction on $|\Pi|$.

- For the base case, use Lemma 7.3.3.

- For the case where Y is a $\triangleright$-structure, we show the sub-case where Y in a negative context, the other case is analogous:

$$\cfrac{\cfrac{\cfrac{\Sigma^-[(Y_1 \triangleright Y_2),(Y_1 \triangleright Y_2)]}{\Sigma^-[(Y_1, Y_1 \triangleright Y_2, Y_2)]}\ \text{Lemma 7.3.13}}{\Sigma^-[(Y_1, Y_1 \triangleright Y_2)]}\ \text{IH}}{\Sigma^-[(Y_1 \triangleright Y_2)]}\ \text{IH}$$

- Case where Y is a $\circ$-structure:

$$\cfrac{\cfrac{\cfrac{\Sigma[\circ(Y_1 \triangleright Y_2),\circ(Y_1 \triangleright Y_2)]}{\Sigma[\circ(Y_1, Y_1 \triangleright Y_2, Y_2)]}\ \text{Lemma 7.3.14}}{\Sigma[\circ(Y_1, Y_1 \triangleright Y_2)]}\ \text{IH}}{\Sigma[\circ(Y_1 \triangleright Y_2)]}\ \text{IH}$$

- Case where Y is a $\bullet$-structure:

$$\cfrac{\cfrac{\cfrac{\Sigma[\bullet(Y_1 \triangleright Y_2),\bullet(Y_1 \triangleright Y_2)]}{\Sigma[\bullet(Y_1, Y_1 \triangleright Y_2, Y_2)]}\ \text{Lemma 7.3.15}}{\Sigma[\bullet(Y_1, Y_1 \triangleright Y_2)]}\ \text{IH}}{\Sigma[\bullet(Y_1 \triangleright Y_2)]}\ \text{IH}$$

Q.E.D.

Once all structural rules of **LBiKt** are shown admissible in **DBiKt**, completeness is straightforward.

Theorem 7.3.17 (Completeness)**.** *For any structures X and Y, if $\vdash_{\mathbf{LBiKt}} \Pi : X \Rightarrow Y$ then $\vdash_{\mathbf{DBiKt}} \Pi' : X \triangleright Y$.*

Proof. By induction on $|\Pi|$, where Π_1' (Π_2') is obtained from Π_1 (Π_2) using the induction hypothesis. As usual, we use dashed lines to indicate that the conclusion is obtained from the premise using the respective lemma.

- Case when Π ends with w_L rule:

$$\cfrac{\begin{matrix}\Pi_1\\ X \Rightarrow Z\end{matrix}}{X, Y \Rightarrow Z}\ w_L \quad\rightsquigarrow\quad \cfrac{\begin{matrix}\Pi_1'\\ X \triangleright Z\end{matrix}}{X, Y \triangleright Z}\ \text{Lemma 7.3.2}$$

- Case when Π ends with w_R rule: analogous to the case for w_L

- Case when Π ends with c_L rule:

$$\cfrac{\begin{matrix}\Pi_1\\ X, Y, Y \Rightarrow Z\end{matrix}}{X, Y \Rightarrow Z}\ c_L \quad\rightsquigarrow\quad \cfrac{\begin{matrix}\Pi_1'\\ X, Y, Y \triangleright Z\end{matrix}}{X, Y \triangleright Z}\ \text{Lemma 7.3.16}$$

- Case when Π ends with c_R rule: analogous to the case for c_L
- Case when Π ends with $\rhd_R$ rule:

$$\dfrac{\overset{\Pi_1}{X, Y \Rightarrow Z}}{X \Rightarrow (Y \rhd Z)}\,\rhd_R \quad\rightsquigarrow\quad \begin{array}{c}\Pi'_1 \\ X, Y \rhd Z \\ \hdashline X \rhd (Y \rhd Z)\end{array}\ \text{Lemma 7.3.8}$$

- Case when Π ends with $\rhd_L$ rule: analogous to the case for $\rhd_R$, using Lemma 7.3.7 instead.
- Case when Π ends with s_L rule:

$$\dfrac{\overset{\Pi_1}{(X_1 \rhd X_2), X_2 \Rightarrow Y_2}}{X_1, X_2 \Rightarrow Y_1, Y_2}\,s_L \quad\rightsquigarrow\quad \begin{array}{c}\Pi'_1 \\ (X_1 \rhd X_2), X_2 \rhd Y_2 \\ \hdashline X_1, X_2 \rhd Y_1, Y_2\end{array}\ \text{Lemma 7.3.5}$$

- Case when Π ends with s_R rule: analogous to the case for s_L, using Lemma 7.3.6 instead.
- Cases when Π ends with $rp_\circ$:

$$\dfrac{\overset{\Pi_1}{X \Rightarrow Y, \circ(W \rhd Z)}}{\bullet(X \rhd Y), W \Rightarrow Z}\,rp_\circ \quad\rightsquigarrow\quad \begin{array}{c}\Pi'_1 \\ X \rhd Y, \circ(W \rhd Z) \\ \hdashline \bullet(X \rhd Y), W \rhd Z\end{array}\ \text{Lemma 7.3.11}$$

$$\dfrac{\overset{\Pi_1}{\bullet(X \rhd Y), W \Rightarrow Z}}{X \Rightarrow Y, \circ(W \rhd Z)}\,rp_\circ \quad\rightsquigarrow\quad \begin{array}{c}\Pi'_1 \\ \bullet(X \rhd Y), W \rhd Z \\ \hdashline X \rhd Y, \circ(W \rhd Z)\end{array}\ \text{Lemma 7.3.12}$$

- Cases when Π ends with $rp_\bullet$: analogous to the cases for $rp_\circ$, using Lemmas 7.3.9 and 7.3.10 instead.
- Case when Π ends with a $\rightarrow_L$ rule:

$$\dfrac{\overset{\Pi_1}{X \Rightarrow A, Y} \qquad \overset{\Pi_2}{X, B \Rightarrow Y}}{X, A \rightarrow B \Rightarrow Y}\,\rightarrow_L \quad\rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi'_1 \\ X \rhd A, Y \\ \hdashline X, A \rightarrow B \rhd A, Y\end{array}\ \text{Lemma 7.3.2} \qquad \begin{array}{c}\Pi'_2 \\ X, B \rhd Y \\ \hdashline X, A \rightarrow B, B \rhd Y\end{array}\ \text{Lemma 7.3.2}}{X, A \rightarrow B \rhd Y}\,\rightarrow_L$$

- Cases when Π ends with $\prec_R$ and all rules for $\vee$, $\wedge$: analogous to the case for $\rightarrow_L$.

- Case when Π ends with a $\Box_L$ rule:

$$\begin{array}{c} \Pi_1 \\ A \Rightarrow (X \rhd Y) \\ \hline \Box A \Rightarrow \circ(X \rhd Y) \end{array}\;\Box_L \quad\rightsquigarrow\quad \begin{array}{cl} \Pi_1' & \\ A \rhd (X \rhd Y) & \\ \hdashline A, \bullet(\Box A \rhd \circ(X \rhd Y)) \rhd (X \rhd Y) & \text{Lemma 7.3.2} \\ \hline \bullet(\Box A \rhd \circ(X \rhd Y)) \rhd (X \rhd Y) & \Box_{L1} \\ \hdashline \bullet(\Box A \rhd \circ(X \rhd Y)), X \rhd Y & \text{Lemma 7.3.6} \\ \hdashline \Box A \rhd \circ(X \rhd Y), \circ(X \rhd Y) & \text{Lemma 7.3.12} \\ \hdashline \Box A \rhd \circ(X \rhd Y) & \text{Lemma 7.3.16} \end{array}$$

- Case when Π ends with a $\blacksquare_L$ rule: analogous to the case for $\Box_L$.
- Case when Π ends with a $\Box_R$ rule:

$$\begin{array}{c} \Pi_1 \\ X \Rightarrow \circ(\emptyset \rhd A) \\ \hline X \Rightarrow \Box A \end{array}\;\Box_R \quad\rightsquigarrow\quad \begin{array}{cl} \Pi_1' & \\ X \rhd \circ(\emptyset \rhd A) & \\ \hdashline X \rhd \Box A, \circ(\emptyset \rhd A) & \text{Lemma 7.3.2} \\ \hline X \rhd \Box A & \Box_R \end{array}$$

- Case when Π ends with a $\blacksquare_R$ rule: analogous to the case for $\Box_R$.
- Case when Π ends with a $\blacklozenge_L$ rule:

$$\begin{array}{c} \Pi_1 \\ \bullet(A \rhd \emptyset) \Rightarrow X \\ \hline \blacklozenge A \Rightarrow X \end{array}\;\blacklozenge_L \quad\rightsquigarrow\quad \begin{array}{cl} \Pi_1' & \\ \bullet(A \rhd \emptyset) \rhd X & \\ \hdashline \bullet(A \rhd \emptyset), \blacklozenge A \rhd X & \text{Lemma 7.3.2} \\ \hline \blacklozenge A \rhd X & \blacklozenge_L \end{array}$$

- Case when Π ends with a $\Diamond_L$ rule: analogous to the case for $\blacklozenge_L$.
- Case when Π ends with a $\blacklozenge_R$ rule:

$$\begin{array}{c} \Pi_1 \\ (X \rhd Y) \Rightarrow A \\ \hline \bullet(X \rhd Y) \Rightarrow \blacklozenge A \end{array}\;\blacklozenge_R \quad\rightsquigarrow\quad \begin{array}{cl} \Pi_1' & \\ (X \rhd Y) \rhd A & \\ \hdashline (X \rhd Y) \rhd \circ(\bullet(X \rhd Y) \rhd \blacklozenge A), A & \text{Lemma 7.3.2} \\ \hline (X \rhd Y) \rhd \circ(\bullet(X \rhd Y) \rhd \blacklozenge A) & \blacklozenge_{R1} \\ \hdashline X \rhd Y, \circ(\bullet(X \rhd Y) \rhd \blacklozenge A) & \text{Lemma 7.3.5} \\ \hdashline \bullet(X \rhd Y), \bullet(X \rhd Y) \rhd \blacklozenge A & \text{Lemma 7.3.11} \\ \hdashline \bullet(X \rhd Y) \rhd \blacklozenge A & \text{Lemma 7.3.16} \end{array}$$

- Case when Π ends with a $\Diamond_R$ rule: analogous to the case for $\blacklozenge_R$.

The cases when Π ends with other logical rules follow immediately from the induction hypothesis and the admissibility of weakening (Lemma 7.3.2). Q.E.D.

Theorem 7.3.18. *For any structures X and Y, $\vdash_{\mathbf{LBiKt}} \Pi : X \Rightarrow Y$ if and only if $\vdash_{\mathbf{DBiKt}} \Pi' : X \triangleright Y$.*

Proof. By Theorems 7.3.1 and 7.3.17. Q.E.D.

7.4 Proof search

In this section we present a proof search strategy for **DBiKt**, which relies on a proof search calculus version of **DBiKt** that we call $\mathbf{DBiKt_1}$. The proof search strategy for $\mathbf{DBiKt_1}$ closely follows the approaches presented in Chapters 5 and 6. Here we emphasize the aspects that are new/different because of the interaction between the tense structures $\circ$ and $\bullet$ and the intuitionistic structure $\triangleright$.

Our proof search strategy proceeds in three stages: saturation, propagation and *realisation*. The saturation phase applies the "static rules" (i.e. those that do not create extra structural connectives) until further application do not lead to any progress. The propagation phase propagates formulae across different structural connectives, while the realisation phase applies the "dynamic rules" (i.e., those that create new structural connectives, e.g., $\rightarrow_R$). In the following, we outline a proof search calculus which eliminates redundancy in proof search.

A context $\Sigma[\,]$ is said to be *headed by a structural connective* # if the topmost symbol in $\Sigma[\,]$ is #. A context $\Sigma[\,]$ is said to be a *factor* of $\Sigma'[\,]$ if $\Sigma[\,]$ is a subcontext of $\Sigma'[\,]$ and $\Sigma[\,]$ is headed by $\triangleright$. We denote with $\widehat{\Sigma[\,]}$ the minimal factor of $\Sigma[\,]$. We write $\widehat{\Sigma[X]}$ to denote the structure $\Sigma_1[X]$, if $\Sigma_1[\,] = \widehat{\Sigma[\,]}$. We define the *top-level* formulae of a structure as: $\{|X|\} = \{A \mid X = (A, Y)$ for some A and $Y\}$. For example, if $\Sigma[] = (A, B \triangleright C, \bullet(D, (E \triangleright F) \triangleright []))$, then $\widehat{\Sigma[G]} = (D, (E \triangleright F) \triangleright G)$, and $\{|D, (E \triangleright F)|\} = \{D\}$.

Let $-\!\!<_{L1}$ and $\rightarrow_{R1}$ denote two new derived rules (section 5.3 contains their derivations):

$$\frac{\Sigma^-[A, A -\!\!< B]}{\Sigma^-[A -\!\!< B]}\ -\!\!<_{L1} \qquad\qquad \frac{\Sigma^+[A \rightarrow B, B]}{\Sigma^+[A \rightarrow B]}\ \rightarrow_{R1}$$

We now define a notion of a *saturated structure*, which is similar to that of a traditional sequent.

Definition 7.4.1. *A structure $X \triangleright Y$ is* saturated *if it satisfies the following:*

1. $\{|X|\} \cap \{|Y|\} = \emptyset$
2. *If $A \wedge B \in \{|X|\}$ then $A \in \{|X|\}$ and $B \in \{|X|\}$*
3. *If $A \wedge B \in \{|Y|\}$ then $A \in \{|Y|\}$ or $B \in \{|Y|\}$*
4. *If $A \vee B \in \{|X|\}$ then $A \in \{|X|\}$ or $B \in \{|X|\}$*
5. *If $A \vee B \in \{|Y|\}$ then $A \in \{|Y|\}$ and $B \in \{|Y|\}$*
6. *If $A \rightarrow B \in \{|X|\}$ then $A \in \{|Y|\}$ or $B \in \{|X|\}$*

7. *If $A \prec B \in \{|Y|\}$ then $A \in \{|Y|\}$ or $B \in \{|X|\}$*

8. *If $A \prec B \in \{|X|\}$ then $A \in \{|X|\}$*

9. *If $A \to B \in \{|Y|\}$ then $B \in \{|Y|\}$*

We define structure membership for any two structures X and Y as follows: $X \in Y$ iff $Y = X, X'$ for some X', modulo associativity and commutativity of comma. For example, $(A \triangleright B) \in (A, (A \triangleright B), \circ(C \triangleright D))$; in this case $X' = A, \circ(C \triangleright D)$. The *realisation of formulae* by a structure X is defined as follows:

- $A \to B$ ($A \prec B$, resp.) is right-realised (resp. left-realised) by X iff there exists a structure $Z \triangleright W \in X$ such that $A \in \{|Z|\}$ and $B \in \{|W|\}$.
- $\Box A$ ($\Diamond A$ resp.) is right-realised (resp. left-realised) by X iff there exists a structure $\circ(Z \triangleright W) \in X$ (resp. $\circ(W \triangleright Z)$) such that $A \in \{|W|\}$.
- $\blacksquare A$ ($\blacklozenge A$ resp.) is right-realised (resp. left-realised) by X iff there exists a structure $\bullet(Z \triangleright W) \in X$ (resp. $\bullet(W \triangleright Z) \in X$) such that $A \in \{|W|\}$.

We say that a structure X is left-realised iff every formula in $\{|X|\}$ with top-level connective $\prec$, $\Diamond$ or $\blacklozenge$ is left-realised by X. Right-realisation of X is defined dually. We say that a structure occurrence X in $\Sigma[X]$ is *propagated* iff no propagation rules are (backwards) applicable to any formula occurrences in X. We define the super-set relation on structures as follows:

- $X_1 \triangleright Y_1 \supset X_0 \triangleright Y_0$ iff $\{|X_1|\} \supset \{|X_0|\}$ or $\{|Y_1|\} \supset \{|Y_0|\}$.
- $\circ(X_1 \triangleright Y_1) \supset \circ(X_0 \triangleright Y_0)$ iff $\bullet(X_1 \triangleright Y_1) \supset \bullet(X_0 \triangleright Y_0)$ iff $X_1 \triangleright Y_1 \supset X_0 \triangleright Y_0$.

Definition 7.4.2. *Given a structure $\Sigma[A]$, we say $\widehat{\Sigma[A]}$ is propagated when its occurrence in $\Sigma[A]$ is propagated, and we say that A is realised by $\widehat{\Sigma[A]}$ when $\widehat{\Sigma[A]} = (X \triangleright Y)$ and either*

- *$A \in \{|X|\}$ and A is left-realised by X, or*
- *$A \in \{|Y|\}$ and A is right-realised by Y*

We now present simple modifications of **DBiKt** to obtain a calculus $\mathbf{DBiKt_1}$ that is amenable to proof search. Our approach follows that of Chapter 5 since we define syntactic restrictions on rules to enforce a search strategy. For example, we stipulate that a structure must be saturated and propagated before child structures can be created using the $\to_R$ rule (see condition 2 of Definition 7.4.3). Termination for $\mathbf{DBiKt_1}$ then follows immediately from the termination arguments for $\mathbf{DBiInt_1}$ (Chapter 5) and **DKt** (Chapter 6), since propagation rules only add new formulae to nodes in nested sequent trees, and the creation of new structures in the nested sequent trees is controlled using the techniques of Chapters 5 and 6.

Definition 7.4.3. *Let $\mathbf{DBiKt_1}$ be the system obtained from* **DBiKt** *with the following changes:*

1. *Add the derived rules* $-\!\!<_{L1}$ *and* $\to_{R1}$.
2. *Restrict rules* $-\!\!<_L$, $\to_R$ *with the following condition: the rule is applicable only if* $\widehat{\Sigma[A\#B]}$ *is saturated and propagated, and* $A\#B$ *is not realised by* $\widehat{\Sigma[A\#B]}$, *for* $\# \in \to, -\!\!<$.
3. *Restrict rules* $\triangleright_{L2}$ *and* $\triangleright_{R2}$ *with the following condition: the rule is applicable only if* $A \notin \{|Y|\}$.
4. *Restrict rules* $\Diamond_L, \Box_R, \blacklozenge_L, \blacksquare_R$ *with the following condition: the rule is applicable only if* $\widehat{\Sigma[\#A]}$ *is saturated and propagated and* $\#A$ *is not realised by* $\widehat{\Sigma[\#A]}$, *for* $\# \in \Diamond, \Box, \blacklozenge, \blacksquare$.
5. *Restrict rules* $\blacksquare_{L2}, \Box_{L2}$ *with the following condition: the rule is applicable only if* $A \notin \{|Y|\}$*:*
6. *Restrict rules* $\Diamond_{R2}, \blacklozenge_{R2}$ *with the following condition: the rule is applicable only if* $A \notin \{|Z|\}$.
7. *Restrict rules* $\to_L$, $-\!\!<_R$, $-\!\!<_{L1}$, $\to_{R1}$, $\triangleright_{L1}$, $\triangleright_{R1}$, $\wedge_L$, $\wedge_R$, $\vee_L$, $\vee_R$ *and all tense propagation rules to the following.*

   Let $\Sigma[X_0]$ *be the conclusion of the rule and let* $\Sigma[X_1]$ *(and* $\Sigma[X_2]$*) be the premise(s). The rule is applicable only if:* $\widehat{\Sigma[X_1]} \supset \widehat{\Sigma[X_0]}$ *and* $\widehat{\Sigma[X_2]} \supset \widehat{\Sigma[X_0]}$.

We now define a mapping from nested sequents to trees of nodes, extending the corresponding mappings of Chapters 5 and 6.

Definition 7.4.4 (Sequents to trees)**.** *Let* Γ *and* Δ *be sets of formulae; let* $a \geq 0$ *for all* $a \in \{n, m, k, l, t, s\}$ *and let:*

$$\begin{array}{rcl}
X_\triangleright & = & (X_1 \triangleright Y_1), \cdots, (X_n \triangleright Y_n) \\
Z_\triangleright & = & (Z_1 \triangleright W_1), \cdots, (Z_m \triangleright W_m) \\
U_\circ & = & \circ(U_1), \cdots, \circ(U_k) \\
V_\bullet & = & \bullet(V_1), \cdots, \bullet(V_l) \\
Q_\circ & = & \circ(Q_1), \cdots, \circ(Q_t) \\
R_\bullet & = & \bullet(R_1), \cdots, \bullet(R_s)
\end{array}$$

Then given the sequent

$$\Xi = X_\triangleright, U_\circ, V_\bullet, \Gamma \triangleright \Delta, Z_\triangleright, Q_\circ, R_\bullet$$

the tree tree(Ξ) *represented by* Ξ *is rooted at node* $\langle \Gamma, \Delta \rangle$, *and has the children as given in Figure 7.4:*

Figure 7.5 gives a proof search strategy for $\mathbf{DBiKt_1}$. Similarly to the proof search strategies of previous chapters, the application of a rule deep inside a sequent can be viewed as focusing on a particular node of the tree. The rules of $\mathbf{DBiKt_1}$ can then

$$\begin{tikzcd}
 & tree(X_1 \triangleright Y_1) \\
 & \vdots \\
 & tree(X_n \triangleright Y_n) \\
 & tree(U_1) \\
 & \vdots \\
 & tree(U_k) \\
 & tree(V_1) \\
 & \vdots \\
 & tree(V_l) \\
\langle \Gamma, \Delta \rangle \arrow[uuuuuuuuur, "\geq"] \arrow[uuuuuuur, "\geq"] \arrow[uuuuuur, "R_\Diamond"] \arrow[uuuur, "R_\Diamond"] \arrow[uuur, "R_\Box^{-1}"] \arrow[ur, "R_\Box^{-1}"] \arrow[dr, "\leq"] \arrow[dddr, "\leq"] \arrow[ddddr, "R_\Box"] \arrow[ddddddr, "R_\Box"] \arrow[dddddddr, "R_\Diamond^{-1}"] \arrow[dddddddddr, "R_\Diamond^{-1}"] & \\
 & tree(Z_1 \triangleright W_1) \\
 & \vdots \\
 & tree(Z_m \triangleright W_m) \\
 & tree(Q_1) \\
 & \vdots \\
 & tree(Q_t) \\
 & tree(R_1) \\
 & \vdots \\
 & tree(R_s)
\end{tikzcd}$$

Figure 7.4: Translation from **DBiKt** sequents to trees.

```
Function Prove (Sequent $\Xi$) : Bool

  1. Let $T = tree(\Xi)$
  2. If the $id$, $\bot_L$, or $\top_R$ rule is applicable to any node in $T$, return $True$
  3. Else if there is some node $\langle \Gamma, \Delta \rangle \in T$ that is not saturated
     (a) Let $\rho$ be the rule corresponding to the requirement of Definition 5.3.2 that
         is not met, and let $\Xi_1$ (and $\Xi_2$) be the premise(s) of $\rho$. Return $\bigwedge Prove(\Xi_i)$.
  4. Else if there is some node $\Theta$ that is not propagated
     (a) Let $\rho$ be the rule corresponding to propagation of formula that is applicable,
         and let $\Xi_1$ be the premise of $\rho$. Return $Prove(\Xi_1)$.
  5. Else if there is some node $\langle \Gamma, \Delta \rangle \in T$ that is not realised:
     (a) If some $C = A \to B \in \Delta$ ($C = A \prec B \in \Gamma$) is not realised then let $\Xi_1$ be the
         premise of the $\to_R$ ($\prec_L$) rule applied to $C \in \Delta$ ($C \in \Gamma$). Return $Prove(\Xi_1)$.
     (b) If some $C = \Diamond A \in \Gamma$ ($C = \blacklozenge A \in \Gamma$) is not realised then let $\Xi_1$ be the premise
         of the $\Diamond_L$ ($\blacklozenge_L$) rule applied to $C = \Diamond A \in \Gamma$ ($C = \blacklozenge A \in \Gamma$). Return $Prove(\Xi_1)$.
     (c) If some $C = \Box A \in \Delta$ ($C = \blacksquare A \in \Delta$) is not realised then let $\Xi_1$ be the
         premise of the $\Box_L$ ($\blacksquare_L$) rule applied to $C = \Box A \in \Delta$ ($C = \blacksquare A \in \Delta$). Return
         $Prove(\Xi_1)$.
  6. Else return $False$
```

Figure 7.5: A proof search strategy for **DBiKt$_1$**

be viewed as operations on the tree encoded in the sequent. In particular, Step 3 saturates a node locally, Step 4 propagates formulae between neighbouring nodes, and Step 5 appends new nodes to the tree. More specifically, Step 5a appends $\leq$ and $\geq$ successors, Step 5b appends $R_\Diamond$ and $R_\Box^{-1}$ successors, and Step 5c appends $R_\Box$ and $R_\Diamond^{-1}$ successors.

Before showing the completeness of *Prove*, we show that it terminates. We define the tense degree of a formula as the maximum number of nested modalities:

$$\begin{array}{rcl} deg(p) & = & 0 \\ deg(A\#B) & = & max(deg(A), deg(B)) \text{ for } \# \in \{\wedge, \vee, \to, \prec\} \\ deg(\#A) & = & 1 + deg(A) \text{ for } \# \in \{\Box, \Diamond, \blacksquare, \blacklozenge\}. \end{array}$$

The degree of a set of formulae is the maximum degree over all its members. We write $sf(A)$ for the subformulae of A, and define the set of subformulae of a set Θ as $sf(\Theta) = \bigcup_{A \in \Theta} sf(A)$. For a sequent Ξ we define $sf(\Xi)$ as the union of the subformulae of all the nodes in $tree(\Xi)$.

The following theorem can be proved by a straightforward combination of the techniques from Chapters 5 and 6.

Theorem 7.4.5. *For any two sets of* `BiKt`*-formulae* Γ *and* Δ, *Prove*$(\Gamma \triangleright \Delta)$ *terminates.*

Proof. Let $m = |sf(\Gamma \triangleright \Delta)|$ and $d = deg(\Gamma \triangleright \Delta) \leq m$. To show that *Prove* terminates, we will argue about the tree $T = tree(X \triangleright Y)$, where $X \triangleright Y$ is the parameter to the most recent recursive call to *Prove*. That is, initially $T = tree(\Gamma \triangleright \Delta)$.

As was the case for $\mathbf{DBiInt_1}$ in Chapter 5, the saturation process for each node in T is bounded by m. Therefore after at most m moves at each node, Step 3 is no longer applicable to this node. Since formulae are only propagated to nodes that do not already contain these formulae, after at most m propagation moves into each node, Step 4 is no longer applicable to this node.

T is finitely branching, since new nodes are only created for unrealised $\rightarrow$, $-\!\!<$, $\Diamond$, $\blacklozenge$, $\Box$, $\blacksquare$ formulae. Therefore after at most m moves at each node, Step 5 is no longer applicable to this node. We will now show that the depth of T is bounded by m^3, extending the argument of Theorem 5.3.9 to the tense connectives.

Notice that every time we create a tense successor node $\Gamma_1 \triangleright \Delta_1$ for some parent node $\Gamma \triangleright \Delta$, we have $deg(\Gamma_1 \cup \Delta_1) = deg(\Gamma \cup \Delta) - 1$. Moreover, every descendant of $\Gamma_1 \triangleright \Delta_1$ will have a tense degree $\leq deg(\Gamma \cup \Delta) - 1$, since there are no rules that can increase the tense degree of a node. Therefore, after $d \leq m$ applications of Steps 5b and 5c on any one branch of T, no tense successor creation rules are applicable.

Secondly, note that there are no tense propagation rules that can increase the bi-intuitionistic degree of a node that we used in the proof of Theorem 7.4.5. Therefore we can use the same argument as that of Theorem 7.4.5 to show that every sequence of $\rightarrow$ and $-\!\!<$ successor creation rules on a branch of T can be at most m^2 long.

Therefore, every branch of T can be at most m^3 long: in the worst case, a node may have an m^2 long chain of bi-intuitionistic descendants, followed by a tense descendant, followed by another m^2 long chain of bi-intuitionistic descendants, and so on at most $d \leq m$ times. Q.E.D.

We will now show the completeness of *Prove* w.r.t. **DBiKt**, using the same method that we used in Chapters 5 and 6. We first give a lemma which shows that if *Prove* returns false for some sequent Ξ, there is no **DBiKt** derivation of that sequent.

Lemma 7.4.6. *Let* Ξ *be a sequent such that every node in* $tree(\Xi)$ *is saturated, realised and propagated. Then* Ξ *is not derivable in* **DBiKt**.

Proof. We prove the statement of the lemma by contradiction. That is, we assume every node in $tree(\Xi)$ is saturated, realised and propagated, and that Ξ is derivable in **DBiKt**. Then there exists a shortest derivation Π of Ξ. We now consider all the rule instances that Π could end with, and in each case show that there is an even shorter **DBiKt** derivation of Ξ, therefore contradicting our assumption that Π was the shortest derivation.

- Π cannot end with the *id* rule, since every node in $tree(\Xi)$ is saturated.

- Suppose Π ends with the $\Box_{L1}$ rule, where $\Xi = \Sigma^-[A, \bullet(\Box A, X \rhd Y)]$ for some $\Sigma^-[]$, and the last rule of Π applies to that particular occurrence of $\bullet(\Box A, X \rhd Y)$ in the context $\Sigma^-[]$. Note that since every node of *tree*(Ξ) is propagated, we know that A is also present at the node where $\bullet(\Box A, X \rhd Y)$ is located. Then Π must be of the form:

$$\cfrac{\begin{array}{c}\Pi_1 \\ \Sigma^-[A, A, \bullet(\Box A, X \rhd Y)]\end{array}}{\Sigma^-[A, \bullet(\Box A, X \rhd Y)]}\ \Box_{L1}$$

  Now, applying Lemma 7.3.3 to Π_1, we obtain a derivation Π_2 of

$$\Sigma^-[A, \bullet(\Box A, X \rhd Y)]$$

  such that $|\Pi_2| = |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\Box_{L1}$.

- All other cases involving tense propagation rules can be treated analogously to $\Box_{L1}$, using height preserving admissibility of formula contraction.

- Suppose Π ends with the $\Diamond_L$ rule, where $\Xi = \Sigma^-[\Diamond A, \circ(X, A \rhd Y)]$ for some $\Sigma^-[]$ and the last rule of Π applies to that particular occurrence of $\Diamond A$ in the context $\Sigma^-[]$. Since every node in *tree*(Ξ) is realised, we know that there is already a structure $\circ(X, A \rhd Y)$ for some X and Y, present at the node where $\Diamond A$ is located. Then Π must be of the form:

$$\cfrac{\begin{array}{c}\Pi_1 \\ \Sigma^-[\Diamond A, \circ(A \rhd \emptyset), \circ(X, A \rhd Y)]\end{array}}{\Sigma^-[\Diamond A, \circ(X, A \rhd Y)]}\ \Diamond_L$$

  By the distribution Lemma 7.3.14, there is a **DBiKt** derivation Π_2 of

$$\Sigma^-[\Diamond A, \circ(X, A, A \rhd Y)]$$

  such that $|\Pi_2| = |\Pi_1|$. Then applying Lemma 7.3.3 to Π_2 gives us a **DBiKt** derivation Π_3 of

$$\Sigma^-[\Diamond A, \circ(X, A \rhd Y)]$$

  such that $|\Pi_3| = |\Pi_2| = |\Pi_1| < |\Pi|$. But this contradicts the assumption that Π is a shortest derivation of Ξ. Therefore Π cannot end with the rule $\Diamond_L$.

- All other cases involving rules $\Box_R$, $\blacklozenge_L$ and $\blacksquare_R$ can be treated analogously to $\Diamond_L$, using the distribution lemmas for $\circ$ and $\bullet$ structures (Lemmas 7.3.14 and 7.3.15) and admissibility of formula contraction.

- Π cannot end with any of the other rules of **DBiKt**: the proof of Lemma 5.3.5 applies to those cases.

Since Π cannot end with any of the rules of **DBiKt**, this obviously contradicts the assumption that it is a derivation in **DBiKt**. Therefore Ξ is not derivable in **DBiInt**.

Q.E.D.

Theorem 7.4.7. *For any X and Y, $Prove(X \triangleright Y) = True$ if and only if $\vdash_{\mathbf{DBiKt}} \Pi : X \triangleright Y$.*

Proof.

- Left-to-right: obvious, since every step of *Prove* is a backwards application of a $\mathbf{DBiKt_1}$ rule, which is either a (possibly restricted) rule of **DBiKt** or a derived rule.
- Right-to-left: we show that if $Prove(X \triangleright Y)$ returns *False* then $X \triangleright Y$ is not derivable in **DBiKt**. Since each rule of **DBiKt** is invertible (Lemma 7.3.4), Steps 2 to 5 of *Prove* preserve provability of the original sequent. If $Prove(X \triangleright Y)$ returns *False*, this can only be the case if Step 6 is reached, i.e., the systematic bottom-up applications of the rules of $\mathbf{DBiKt_1}$ produce a sequent such that every node in the tree of the sequent is saturated, realised, and propagated. By Lemma 7.4.6, such a sequent would not be derivable in **DBiKt**, and since all other steps of *Prove* preserve derivability, it follows that $X \triangleright Y$ is not derivable either in **DBiKt**.

Q.E.D.

Theorem 7.4.8. `BiKt` *is decidable.*

Proof. By Theorems 7.3.1, 7.3.17, 7.4.7 and 7.4.5. Q.E.D.

A prototype implementation of $\mathbf{DBiKt_1}$ is available online at `http://users.cecs.anu.edu.au/~linda/dbikt.zip`.

7.5 Semantics

Our semantics for `BiKt` extend Rauszer's [100] Kripke-style semantics for `BiInt` (recall Section 2.2.4.4) by clauses for the tense logic connectives.

A `BiKt` *frame* is a tuple $\langle W, \leq, R_\Diamond, R_\Box \rangle$ where W is a non-empty set (of possible worlds) and $\leq \subseteq (W \times W)$ is a reflexive and transitive binary relation over W, and each of $R_\Diamond$ and $R_\Box$ are arbitrary binary relations over W with the following *frame conditions*:

F1$\Diamond$ if $x \leq y$ & $xR_\Diamond z$ then $\exists w.\ yR_\Diamond w$ & $z \leq w$

F2$\Box$ if $xR_\Box y$ & $y \leq z$ then $\exists w.\ x \leq w$ & $wR_\Box z$.

A `BiKt` *model* extends a `BiKt` frame with a mapping V from *Atoms* to 2^W obeying *persistence:* $\forall v \geq w.\ w \in V(p) \Rightarrow v \in V(p)$.

Definition 7.5.1 (Forcing). *Given a model $\langle W, \leq, R_\Diamond, R_\Box, V \rangle$, we say that $w \in W$* forces *p if $w \in V(p)$, and write this as $w \Vdash p$. We write $w \not\Vdash p$ to mean $(not)(w \Vdash p)$; that is, $\exists v \geq w.\ v \notin V(p)$, and say that w* rejects *p. The relation $\Vdash$ is then extended to the verum and falsum constants and compound formulae as given in Figure 7.6.*

$w \Vdash \top$		for every $w \in W$
$w \Vdash \bot$		for no $w \in W$
$w \Vdash A \wedge B$	iff	$w \Vdash A$ and $w \Vdash B$
$w \Vdash A \vee B$	iff	$w \Vdash A$ or $w \Vdash B$
$w \Vdash A \rightarrow B$	iff	if $\forall v \geq w.\ v \Vdash A$ then $v \Vdash B$
$w \Vdash A \prec B$	iff	$\exists v \leq w.\ v \Vdash A$ and $v \nVdash B$
$w \Vdash \Diamond A$	iff	$\exists v.\ wR_{\Diamond}v$ and $v \Vdash A$
$w \Vdash \Box A$	iff	$\forall z.\forall v.$ if $w \leq z$ and $zR_{\Box}v$ then $v \Vdash A$
$w \Vdash \blacklozenge A$	iff	$\exists v.\ wR_{\Box}^{-1}v$ and $v \Vdash A$
$w \Vdash \blacksquare A$	iff	$\forall z.\forall v.$ if $w \leq z$ and $zR_{\Diamond}^{-1}v$ then $v \Vdash A$

Figure 7.6: Semantics of BiKt

Similarly to the case of bi-intuitionistic logic, we define the concept of rejection.

Definition 7.5.2 (Rejection). *Given a model $\langle W, \leq, R_{\Diamond}, R_{\Box}, V\rangle$, a world $w \in W$ and a* BiKt *formula A, we say that w* rejects *A if $w \nVdash A$.*

Now we can extend the concepts of forcing and rejection to sets of formuale.

Definition 7.5.3. *Given a model $\langle W, \leq, R_{\Diamond}, R_{\Box}, V\rangle$, a world $w \in W$ and sets of* BiKt *formulae Γ and Δ, we write*

1. *$w \Vdash \Gamma$ iff $\forall A \in \Gamma . w \Vdash A$*
2. *$w \dashVv \Delta$ iff $\forall A \in \Delta . w \nVdash A$*

As usual, a BiKt-formula A is BiKt-*valid* if it is satisfied by every world in every Kripke model. A nested sequent $X \rhd Y$ is BiKt-valid if its formula translation is BiKt-valid.

Our semantics differ from those of Simpson [104] and Ewald [39] because we use two modal accessibility relations instead of one. In our calculi, there is no direct relationship between $\Diamond$ and $\Box$ (or $\blacklozenge$ and $\blacksquare$), but $\Diamond$ and $\blacksquare$ are a residuated pair, as are $\blacklozenge$ and $\Box$. Semantically, this corresponds to $R_{\blacklozenge} = R_{\Box}^{-1}$ and $R_{\blacksquare} = R_{\Diamond}^{-1}$; therefore the clauses in Figure 7.6 are couched in terms of $R_{\Diamond}$ and $R_{\Box}$ only. Our frame conditions F1$\Diamond$ and F2$\Box$ are also used by Simpson whose F2 captures the "persistence of being seen by" [104, page 51] while for us F2$\Box$ is simply the "persistence of $\blacklozenge$".

We now show that **LBiKt** is sound with respect to BiKt semantics.

Lemma 7.5.4. *If $\Rightarrow A$ is* **LBiKt***-derivable then A is* BiKt*-valid.*

Proof. We show that for each rule of **LBiKt**, if its premise sequents are BiKt-valid then its conclusion is also BiKt-valid. The cases for rules relating to the bi-intuitionistic fragment are straightforward; we show the cases for the tense connectives. In each case, we assume that the formula-translation of the premise is valid and show that the formula-translation of the conclusion is valid. Recall that Figure 7.1 gives the formula-translation of **LBiKt** sequents. For the rules $rp''_\circ$ and $rp''_\bullet$, we actually prove soundness of the following two simplified versions, from which the actual rules can be easily derived using s_L, s_R, $\triangleright_L$ and $\triangleright_R$:

$$\frac{\bullet X \Rightarrow Y}{X \Rightarrow \circ Y}\, rp''_\circ \qquad \frac{\circ X \Rightarrow Y}{X \Rightarrow \bullet Y}\, rp''_\bullet$$

- Rule $rp''_\circ$ (top-to-bottom): we assume that $\tau^-(\bullet X) \to \tau^+(Y)$ is valid and show that $\tau^-(X) \to \tau^+(\circ Y)$ is valid. That is, we assume that $\blacklozenge\tau^-(X) \to \tau^+(Y)$ is valid and show that $\tau^-(X) \to \Box\tau^+(Y)$ is valid. Since $\blacklozenge\tau^-(X) \to \tau^+(Y)$ is valid, we know that:

$$\forall\langle W, \leq, R_\Diamond, R_\Box, V\rangle \forall w \in W.\text{ if } w \Vdash \blacklozenge\tau^-(X) \text{ then } w \Vdash \tau^+(Y) \tag{7.5.1}$$

  To show that $\tau^-(X) \to \Box\tau^+(Y)$ is valid, we will use proof by contradiction. That is, we assume $\tau^-(X) \to \Box\tau^+(Y)$ is falsifiable, which gives us a `BiKt` model $\langle W, \leq, R_\Diamond, R_\Box, V\rangle$ and a world $u \in W$ such that:

$$u \Vdash \tau^-(X) \tag{7.5.2}$$
$$u \nVdash \Box\tau^+(Y) \tag{7.5.3}$$

  (7.5.3) means that there exist worlds $z, v \in W$ such that:

$$u \leq z \tag{7.5.4}$$
$$zR_\Box v \tag{7.5.5}$$
$$v \nVdash \tau^+(Y) \tag{7.5.6}$$

  Now, by the persistence property of `BiKt` frames, (7.5.2) and (7.5.4) imply:

$$z \Vdash \tau^-(X) \tag{7.5.7}$$

  Secondly, (7.5.7) and (7.5.5) imply that $v \Vdash \blacklozenge\tau^-(X)$. However, we already have $v \nVdash \tau^+(Y)$ by (7.5.6), which contradicts (7.5.1). Therefore our assumption that $\tau^-(X) \to \Box\tau^+(Y)$ is falsifiable was incorrect, and indeed $\tau^-(X) \to \Box\tau^+(Y)$ is valid.

- Rule $rp''_\circ$ (bottom-to-top): we assume that $\tau^-(X) \to \tau^+(\circ Y)$ is valid and show that $\tau^-(\bullet X) \to \tau^+(Y)$ is valid. That is, we assume that $\tau^-(X) \to \Box\tau^+(Y)$ is valid and show that $\blacklozenge\tau^-(X) \to \tau^+(Y)$ is valid. The rest of the proof is similar to the case for $rp''_\circ$ (top-to-bottom): we attempt to create a countermodel for $\blacklozenge\tau^-(X) \to \tau^+(Y)$ which contains worlds w, u such that $w \Vdash \blacklozenge\tau^-(X)$ and $w \nVdash \tau^+(Y)$, and $vR_\Box w$ and $v \Vdash \tau^-(X)$. In this purported countermodel, we also get $v \Vdash \Box\tau^+(Y)$ from the assumed validity of $\tau^-(X) \to \Box\tau^+(Y)$, which gives $w \Vdash \tau^+(Y)$, thus contradicting $w \nVdash \tau^+(Y)$.

- Rule $rp''_\bullet$: analogous to $rp''_\circ$.

- Rule $\Box_L$: we prove the lemma for the following simplified rule:

$$\frac{A \Rightarrow Z}{\Box A \Rightarrow \circ Z}\ \Box''_L$$

Note that the actual rule $\Box_L$ is an instance of $\Box''_L$ with $Z = (X \triangleright Y)$.

We assume that $A \rightarrow \tau^+(Z)$ is valid and show that $\Box A \rightarrow \Box\tau^+(Z)$ is valid. Since $A \rightarrow \tau^+(Z)$ is valid, we know that:

$$\forall\langle W, \leq, R_\Diamond, R_\Box, V\rangle \forall w \in W.\ \text{if } w \Vdash A \text{ then } w \Vdash \tau^+(Z) \tag{7.5.8}$$

To show that $\Box A \rightarrow \Box\tau^+(Z)$ is valid, we will use proof by contradiction. That is, we assume $\Box A \rightarrow \Box\tau^+(Z)$ is falsifiable, which gives us a `BiKt` model $\langle W, \leq, R_\Diamond, R_\Box, V\rangle$ and a world $u \in W$ such that:

$$u \Vdash \Box A \tag{7.5.9}$$

$$u \not\Vdash \Box\tau^+(Z) \tag{7.5.10}$$

(7.5.9) gives us that:

$$\forall z.\forall v.\ \text{if } u \leq z \text{ and } zR_\Box v \text{ then } v \Vdash A \tag{7.5.11}$$

While (7.5.10) gives us that:

$$\exists z'.\exists v'.u \leq z' \text{ and } z'R_\Box v' \text{ and } v' \not\Vdash \tau^+(Z) \tag{7.5.12}$$

By (7.5.11), we also have $v' \Vdash A$. But this, together with (7.5.12) contradicts (7.5.8). Therefore our assumption that $\Box A \rightarrow \Box\tau^+(Z)$ is falsifiable was incorrect, and indeed $\Box A \rightarrow \Box\tau^+(Z)$ is valid.

- The cases for rules $\blacksquare_L$, $\Diamond_R$ and $\blacklozenge_R$ are analogous to the case for $\Box_L$.

- The cases for rules $\Diamond_L$, $\blacklozenge_L$, $\Box_R$, $\blacksquare_R$ are obvious.

Q.E.D.

We now show the completeness of **DBiKt** with respect to the semantics of `BiKt`.

Theorem 7.5.5. *If a* `BiKt`*-formula A is* `BiKt`*-valid then $\triangleright A$ is* **DBiKt***-derivable.*

Proof. To prove completeness, we assume that $\triangleright A$ is not derivable, that is, there is no **DBiKt** derivation of $\triangleright A$, and extract a countermodel for A. We follow the usual counter-model construction technique for intuitionistic and tense logics; the non-trivial addition is showing that the resulting models satisfy the frame conditions F1$\Diamond$ and F2$\Box$. Specifically, we show that if $\triangleright A$ is not **DBiKt**-derivable, then there exists a `BiKt`-model that makes A false.

Since we know by Theorem 7.4.7 that *Prove* is complete w.r.t. **DBiKt**, it suffices to show that we can extract such a model from a failed *Prove* attempt. Consider the tree

T such that $T = tree(\Xi)$ and $Prove(\Xi) = False$, and $Prove(\Xi)$ was the last recursive call invoked by $Prove(\triangleright A)$. We will now show how we can turn T into a `BiKt` model. We say that some tree T' rooted at $\langle \Gamma, \Delta \rangle$ has a `BiKt` model if there exists a model $M = \langle W, \leq, R_\Diamond, R_\Box, V \rangle$ such that there exists a world $w \in W$ such that $w \Vdash \Gamma$ and $w \dashv\!\mid \Delta$.

We proceed by induction on the height of T. For the base case, T consists of one leaf node $\langle \Gamma, \Delta \rangle$. We create the following model:

- Let $W = \{w\}$
- Let $\leq = \{(w, w)\}$
- Let $R_\Diamond = \emptyset$
- Let $R_\Box = \emptyset$
- Let $V(p) = \begin{cases} \{w\} & \text{if } p \in \Gamma \\ \emptyset & \text{otherwise} \end{cases}$

The valuation V is defined so that every atom in Γ is forced by w, and every other atom, including those in Δ, is rejected by w. Since we know that every node in T is saturated, it follows from Definitions 7.4.1 and 7.5.3 that every bi-intuitionistic formula in Γ (Δ) is forced (rejected) respectively. Moreover, since every node in T is realised and $\langle \Gamma, \Delta \rangle$ has no successors in T, we know from Definition 7.4.2 that Γ and Δ do not contain any tense formulae. Therefore we have $w \Vdash \Gamma$ and $w \dashv\!\mid \Delta$.

For the induction hypothesis, we assume that for every tree T' of height $< k$, there exists a `BiKt` model for the root of T'. Now consider a tree of height k. Denote its root node with $\langle \Gamma, \Delta \rangle$.

- Let $W = \{w_0\}$
- Let $\leq = \emptyset$
- Let $R_\Diamond = \emptyset$
- Let $R_\Box = \emptyset$
- Let $V(p) = \begin{cases} \{w_0\} & \text{if } p \in \Gamma \\ \emptyset & \text{otherwise} \end{cases}$

1. For each child node $\langle \Gamma_1, \Delta_1 \rangle$ that is a $\geq$ successor of $\langle \Gamma, \Delta \rangle$ in T: By the induction hypothesis, we know that the tree rooted at $u = \langle \Gamma_1, \Delta_1 \rangle$ has a model $M = \langle W', \leq', R'_\Diamond, R'_\Box, V' \rangle$. We connect w_0 to this model as follows:
   (a) Let $W = W \cup W'$
   (b) Let $\leq$ be the reflexive, transitive closure of $\leq \cup \leq' \cup \{(u \leq w_0)\}$
   (c) Let $R_\Diamond = R_\Diamond \cup R'_\Diamond$
   (d) Let $R_\Box = R_\Box \cup R'_\Box$

   (e) Let $V = V \cup V'$

2. For each child node $\langle \Gamma_1, \Delta_1 \rangle$ that is a $\leq$ successor of $\langle \Gamma, \Delta \rangle$ in T: perform an analogous construction to Step 1, but let $\leq$ be the reflexive, transitive closure of $\leq \cup \leq' \cup \{(w_0 \leq u)\}$ instead.

3. For each child node $\langle \Gamma_1, \Delta_1 \rangle$ that is a $R_\Diamond$ successor of $\langle \Gamma, \Delta \rangle$ in T: By the induction hypothesis, we know that the tree rooted at $u = \langle \Gamma_1, \Delta_1 \rangle$ has a model $M = \langle W', \leq', R'_\Diamond, R'_\Box, V' \rangle$. We connect w_0 to this model as follows:

   (a) Let $W = W \cup W'$

   (b) Let $\leq$ be the reflexive, transitive closure of $\leq \cup \leq'$

   (c) Let $R_\Diamond = R_\Diamond \cup R'_\Diamond \cup \{(w_0 R_\Diamond u)\}$

   (d) Let $R_\Box = R_\Box \cup R'_\Box$

   (e) Let $V = V \cup V'$

4. For each child node $\langle \Gamma_1, \Delta_1 \rangle$ that is a $R_\Box^{-1}$ successor of $\langle \Gamma, \Delta \rangle$ in T: perform an analogous construction to Step 3, but let $R_\Box = R_\Box \cup R'_\Box \cup \{(u R_\Box w_0)\}$ instead.

5. For each child node $\langle \Gamma_1, \Delta_1 \rangle$ that is a $R_\Box$ successor of $\langle \Gamma, \Delta \rangle$ in T: By the induction hypothesis, we know that the tree rooted at $u = \langle \Gamma_1, \Delta_1 \rangle$ has a model $M = \langle W', \leq', R'_\Diamond, R'_\Box, V' \rangle$. We connect w_0 to this model as follows:

   (a) Let $W = W \cup W'$

   (b) Let $\leq$ be the reflexive, transitive closure of $\leq \cup \leq'$

   (c) Let $R_\Diamond = R_\Diamond \cup R'_\Diamond$

   (d) Let $R_\Box = R_\Box \cup R'_\Box \cup \{(w_0 R_\Box u)\}$

   (e) Let $V = V \cup V'$

6. For each child node $\langle \Gamma_1, \Delta_1 \rangle$ that is a $R_\Diamond^{-1}$ successor of $\langle \Gamma, \Delta \rangle$ in T: perform an analogous construction to Step 5, but let $R_\Diamond = R_\Diamond \cup R'_\Diamond \cup \{(u R_\Diamond w_0)\}$ instead.

From the above construction, notice that every node in the nested sequent tree becomes a world in the model. Now we need to show that the resulting model M is a model for $\langle \Gamma, \Delta \rangle$, which involves showing that:

1. $w \Vdash \Gamma$

2. $w \dashv\!\!| \Delta$

3. M satisfies persistence and reverse persistence

4. M satisfies the the frame conditions F1$\Diamond$ and F2$\Box$

We first prove (1) and (2) by simultaneous induction on the length of formulae in Γ and Δ. For each such formula $A \in \Gamma \cup \Delta$:

Base case If A is an atomic formula p, then the valuation ensures that $w \Vdash p$ for all $p \in \Gamma$. Since we know that each node is saturated, we also know that $\Gamma \cap \Delta = \emptyset$ from Definition 7.4.1, then we also have that $w \dashv\!\!| \, q$ for all $q \in \Delta$.

Inductive case

- If $A = B \wedge C \in \Gamma$: Then by Definition 7.4.1, we have that $B \in \Gamma$ and $C \in \Gamma$, and therefore by the induction hypothesis we have that $w \Vdash B$ and $w \Vdash C$. Therefore, by Definition 7.5.1, we have $w \Vdash B \wedge C$ as required.
- If $A = B \wedge C \in \Delta$: Then by Definition 7.4.1, we have that either $B \notin \Delta$ or $C \notin \Delta$, and therefore by the induction hypothesis we have that either $w \dashv\!\!| \, B$ or $w \dashv\!\!| \, C$. Therefore, by Definition 7.5.1, we have $w \dashv\!\!| \, B \wedge C$ as required.
- If $A = B \vee C \in \Gamma$: similar to the case for $A = B \wedge C$.
- If $A = B \vee C \in \Delta$: similar to the case for $A = B \wedge C$.
- If $A = B \to C \in \Gamma$: similar to the case for $A = B \wedge C$.
- If $A = B \to C \in \Delta$: Then by Definition 7.4.2 using the rule $\to_R$, we have that there exists a $\leq$ successor $\langle \Gamma_1, \Delta_1 \rangle$ of $\langle \Gamma, \Delta \rangle$ in T, and $B \in \Gamma_1$ and $C \in \Delta_1$. Moreover, at Step 2 above, we have created a world u such that $u \Vdash \Gamma_1$ and $u \dashv\!\!| \, \Delta_1$ and $w \leq u$. Therefore, by Definition 7.5.1, we have $w \dashv\!\!| \, B \to C$ as required.
- If $A = B \multimap\!\!< C \in \Gamma$: similar to the case for $A = B \to C \in \Delta$.
- If $A = B \multimap\!\!< C \in \Delta$: similar to the case for $A = B \wedge C$.
- If $A = \Box B \in \Gamma$: By Definition 7.5.1 and Figure 7.6, to show that $w \Vdash \Box B$, we need to show that $\forall z. \forall v.$ if $w \leq z$ and $zR_\Box v$ then $v \Vdash B$.

  1. By reflexivity, we have $w \leq w$, so we first show that $\forall v.$ if $wR_\Box v$ then $v \Vdash B$. First, note that by Definition 7.4.2 using the propagation rule $\Box_{L2}$, we have that every $R_\Box$ successor $\langle \Gamma_1, \Delta_1 \rangle$ of $\langle \Gamma, \Delta \rangle$ in T has $B \in \Gamma_1$. Moreover, at Step 5 above, we have connected the world $v = \langle \Gamma_1, \Delta_1 \rangle$ to w using $wR_\Box v$, and by the induction hypothesis we have $v \Vdash B$.
  2. We also need to show that for other every $\leq$-successor $z = \langle \Gamma', \Delta' \rangle$ of $\langle \Gamma, \Delta \rangle$, for every $R_\Box$ successor $v = \langle \Gamma_2, \Delta_2 \rangle$ of z, it is the case that $B \in \Gamma_2$. Since we know that every node is propagated, rule $\triangleright_{L2}$ ensures that if $\Box B \in \Gamma$ then $\Box B \in \Gamma'$. Additionally, propagation rule $\Box_{L2}$ ensures that $\langle \Gamma_2, \Delta_2 \rangle$ has $B \in \Gamma_2$. Again, at Step 5 above, applied to $z = \langle \Gamma', \Delta' \rangle$ we have connected the world $v = \langle \Gamma_2, \Delta_2 \rangle$ to z using $zR_\Box v$, and by the induction hypothesis we have $v \Vdash B$.

  Therefore, by Definition 7.5.1, we have $w \Vdash \Box B$.
- If $A = \Box B \in \Delta$: By Definition 7.5.1 and Figure 7.6, to show that $w \dashv\!\!| \, \Box B$, we need to show that $\exists z. \exists v. w \leq z$ and $zR_\Box v$ and $v \dashv\!\!| \, B$. We use reflexivity and take $z = w$, so we need to show $\exists v.\ wR_\Box v$ and $v \dashv\!\!| \, B$. By Definition 7.4.2 using the $\Box_R$ rule, we have that there exists an $R_\Box$ successor $\langle \Gamma_1, \Delta_1 \rangle$ of

$\langle \Gamma, \Delta \rangle$ in T, such that $B \in \Delta_1$. Moreover, at Step 5 above, we have connected the world $v = \langle \Gamma_1, \Delta_1 \rangle$ to w using $wR_\Box v$, and by the induction hypothesis we have $v \dashv\!\mid B$. Therefore, by Definition 7.5.1, we have $w \dashv\!\mid \Box B$.

- If $A = \blacksquare B \in \Gamma$: Similar to the case for $A = \Box B \in \Gamma$.
- If $A = \blacksquare B \in \Delta$: Similar to the case for $A = \Box B \in \Delta$.
- If $A = \Diamond B \in \Gamma$: Similar to the case for $A = \Box B \in \Delta$.
- If $A = \Diamond B \in \Delta$: Similar to the case for $A = \Box B \in \Gamma$.
- If $A = \blacklozenge B \in \Gamma$: Similar to the case for $A = \Diamond B \in \Gamma$.
- If $A = \blacklozenge B \in \Delta$: Similar to the case for $A = \Diamond B \in \Delta$.

Showing that the resulting models satisfy persistence and reverse persistence is easy: by Definition 7.4.2, we know that the propagation rules $\triangleright_{L1}$, $\triangleright_{R1}$, $\triangleright_{L2}$ and $\triangleright_{R2}$ have been applied to every node they are applicable to. Rules $\triangleright_{L1}$ and $\triangleright_{L2}$ guarantee forward persistence, while rules $\triangleright_{R1}$ and $\triangleright_{R2}$ guarantee reverse persistence.

We now need to show that resulting models satisfy the frame conditions F1$\Diamond$ and F2$\Box$; we give the case for F1$\Diamond$ and the other is analogous. We need to consider all fragments of models constructed that contain three worlds x, y and z such that $x \leq y$ and $xR_\Diamond z$. Then we need to show that there exists a w such that $yR_\Diamond w$ and $z \leq w$. We will take z to be the required w, which means we need to show that $yR_\Diamond z$ and $z \leq z$. The latter follows immediately because $\leq$ is reflexive. The former demands the following:

1. If $y \not\Vdash \Diamond B$ then $z \not\Vdash B$
2. If $z \Vdash \blacksquare A$ then $y \Vdash A$

According to our construction, there are three ways of obtaining worlds x, y and z such that $x \leq y$ and $xR_\Diamond z$. We now show that (1) and (2) above hold in each such case:

- The world y has an $\geq$-successor x, and the world x has an $R_\Diamond$ successor z. That is, the sequent fragment corresponding to the model fragment is of the form $\Sigma[(\circ(Z_1 \triangleright Z_2), X_1 \triangleright X_2), Y_1 \triangleright Y_2]$, where $x = \langle X_1, X_2 \rangle$, $y = \langle Y_1, Y_2 \rangle$ and $z = \langle Z_1, Z_2 \rangle$.

  1. We need to show that if $\Diamond B \in Y_2$ then $B \in Z_2$; then our construction above will ensure that $z \not\Vdash B$.

     Recall that by Definition 7.4.2, we know that all propagation rules have been applied to every node they are applicable to. Then the following propagation rules give us the required: rule $\triangleright_{R2}$ ensures that $\Diamond B$ is propagated to node x, and rule $\Diamond_{R2}$ ensures that B is propagated to node Z. We summarise the propagations in the following derivation fragment:

$$\dfrac{\dfrac{\Sigma[(\circ(Z_1 \triangleright Z_2, B), X_1 \triangleright X_2, \Diamond B), Y_1 \triangleright Y_2, \Diamond B]}{\Sigma[(\circ(Z_1 \triangleright Z_2), X_1 \triangleright X_2, \Diamond B), Y_1 \triangleright Y_2, \Diamond B]}\ \Diamond_{R2}}{\Sigma[(\circ(Z_1 \triangleright Z_2), X_1 \triangleright X_2), Y_1 \triangleright Y_2, \Diamond B]}\ \triangleright_{R2}$$

2. We need to show that if $\blacksquare A$ in Z_1 then $A \in Y_1$. We achieve this using the propagation rules $\blacksquare_{L1}$ and $\triangleright_{L1}$, as illustrated in the following derivation fragment:

$$\dfrac{\dfrac{\Sigma[A, (A, \circ(\blacksquare A, Z_1 \triangleright Z_2), X_1 \triangleright X_2), Y_1 \triangleright Y_2]}{\Sigma[(A, \circ(\blacksquare A, Z_1 \triangleright Z_2), X_1 \triangleright X_2), Y_1 \triangleright Y_2]}\triangleright_{L1}}{\Sigma[(\circ(\blacksquare A, Z_1 \triangleright Z_2), X_1 \triangleright X_2), Y_1 \triangleright Y_2]}\blacksquare_{L1}$$

- The world x has a $\leq$ successor y and a $R_\Diamond$ successor z. That is, the sequent fragment corresponding to the model fragment is of the form $\Sigma[\circ(Z_1 \triangleright Z_2), X_1 \triangleright X_2, (Y_1 \triangleright Y_2)]$. Again, we use propagation rules as illustrated in the following derivation fragments:

1. We need to show that if $\Diamond B \in Y_2$ then $B \in Z_2$:

$$\dfrac{\dfrac{\Sigma[\circ(Z_1 \triangleright Z_2, B), X_1 \triangleright X_2, (Y_1 \triangleright Y_2, \Diamond B), \Diamond B]}{\Sigma[\circ(Z_1 \triangleright Z_2), X_1 \triangleright X_2, (Y_1 \triangleright Y_2, \Diamond B), \Diamond B]}\Diamond_{R2}}{\Sigma[\circ(Z_1 \triangleright Z_2), X_1 \triangleright X_2, (Y_1 \triangleright Y_2, \Diamond B)]}\triangleright_{R1}$$

2. We need to show that if $\blacksquare A$ in Z_1 then $A \in Y_1$:

$$\dfrac{\dfrac{\Sigma[A, \circ(\blacksquare A, Z_1 \triangleright Z_2), X_1 \triangleright X_2, (A, Y_1 \triangleright Y_2)]}{\Sigma[A, \circ(\blacksquare A, Z_1 \triangleright Z_2), X_1 \triangleright X_2, (Y_1 \triangleright Y_2)]}\triangleright_{L2}}{\Sigma[\circ(\blacksquare A, Z_1 \triangleright Z_2), X_1 \triangleright X_2, (Y_1 \triangleright Y_2)]}\blacksquare_{L1}$$

- The world z has a $R_\Box^{-1}$ successor x, and x has a $\leq$ successor y. That is, the sequent fragment corresponding to the model fragment is of the form $\Sigma[Z_1 \triangleright Z_2, \bullet(X_1 \triangleright X_2, (Y_1 \triangleright Y_2))]$. Again, we use propagation rules as illustrated in the following derivation fragments:

1. We need to show that if $\Diamond B \in Y_2$ then $B \in Z_2$:

$$\dfrac{\dfrac{\Sigma[Z_1 \triangleright Z_2, B, \bullet(X_1 \triangleright X_2, (Y_1 \triangleright Y_2, \Diamond B), \Diamond B)]}{\Sigma[Z_1 \triangleright Z_2, \bullet(X_1 \triangleright X_2, (Y_1 \triangleright Y_2, \Diamond B), \Diamond B)]}\Diamond_{R1}}{\Sigma[Z_1 \triangleright Z_2, \bullet(X_1 \triangleright X_2, (Y_1 \triangleright Y_2, \Diamond B))]}\triangleright_{R1}$$

2. We need to show that if $\blacksquare A$ in Z_1 then $A \in Y_1$:

$$\dfrac{\dfrac{\Sigma[Z_1, \blacksquare A \triangleright Z_2, \bullet(A, X_1 \triangleright X_2, (A, Y_1 \triangleright Y_2))]}{\Sigma[Z_1, \blacksquare A \triangleright Z_2, \bullet(A, X_1 \triangleright X_2, (Y_1 \triangleright Y_2))]}\triangleright_{L2}}{\Sigma[Z_1, \blacksquare A \triangleright Z_2, \bullet(X_1 \triangleright X_2, (Y_1 \triangleright Y_2))]}\blacksquare_{L2}$$

Q.E.D.

7.6 Modularity, extensions and classicality

We now show how we can obtain Ewald's intuitionistic tense logic IKt [39], Simpson's intuitionistic modal logic IK [104] and regain classical tense logic `Kt`. We also discuss extensions of **DBiKt** with axioms T, 4 and B but they do not correspond semantically to reflexivity, transitivity and symmetry [104].

7.6.1 Modularity

Notice that the semantics of `BiKt` extends modularly that of intuitionistic or dual intuitionistic logic (i.e., the R components in the Kripke frames are irrevelant). So semantically, any formula composed using only (dual-)intuitionistic connectives is (dual-)intuitionistically valid iff it is `BiKt`-valid. A analogous observation applies to other fragments obtained by, say, restricting to purely modal or purely tense connectives. We show a couple of modularity results for intuitionistic logic and a version of intuitionistic modal logic.

We denote with `IntK` the logic extending `Int` with modal operators. The notions of `Int`-validity and `IntK`-validity are defined in the obvious way.

A nested sequent is **purely modal** if contains no occurrences of $\bullet$ nor its formula translates $\blacksquare$ and $\blacklozenge$. We write **DInt** for the sub-system of **DBiKt** containing only the rules *id*, the logical rules for intuitionistic connectives, and the propagation rules for $\triangleright$. The logical system **DIntK** is obtained by adding to **DInt** the deep introduction rules for $\Box$ and $\Diamond$, and the propagation rules $\Box_{L2}$ and $\Diamond_{R2}$.

Observe that in **DBiKt**, the only rules that create $\bullet$ upwards are $\blacklozenge_L$ and $\blacksquare_R$. Thus in every **DBiKt**-derivation Π of an `IntK` formula, the internal sequents in Π are purely modal, and hence Π is also a **DIntK**-derivation. The following results follow immediately from the equivalence of **LBiKt** and **DBiKt** and these observations.

Theorem 7.6.1. *An* `Int` *(resp.* `IntK`*) formula A is* `Int` *(resp.* `IntK`*) valid iff* $\triangleright A$ *is derivable in* **DInt** *(resp.* **DIntK***).*

7.6.2 Obtaining Ewald's IKt

In order to obtain Ewald's IKt [39], semantically, we need to collapse $R_\Diamond$ and $R_\Box$ into one temporal relation R and leave out our semantic clause for $-\!\!<$. That is, we need to add the following conditions to the basic semantics: $R_\Diamond \subseteq R_\Box$ and $R_\Box \subseteq R_\Diamond$. Proof theoretically, this is captured by extending **LBiKt** with the structural rules:

$$\frac{X \Rightarrow \bullet(Y \triangleright \emptyset) \triangleright \bullet(\emptyset \triangleright Z)}{X \Rightarrow \bullet(Y \triangleright Z)}\ \bullet\triangleright_R \qquad \frac{X \Rightarrow \circ(Y \triangleright \emptyset) \triangleright \circ(\emptyset \triangleright Z)}{X \Rightarrow \circ(Y \triangleright Z)}\ \circ\triangleright_R$$

Simpson's intuitionistic modal logic IK [104] can then be obtained from Ewald's system by restricting the language to the modal fragment.

A `BiKt`-frame is an *E-frame* if $R_\Box = R_\Diamond$. A formula A is *E-valid* if it is true in all worlds of every E-model. An IKt formula A is a theorem of IKt iff it is E-valid [39].

We now show that the rules $\circ\rhd_R$ and $\bullet\rhd_R$ are sound for E-frames; for simplicity we replace the structures Y and Z with the atoms p and q.

Lemma 7.6.2. *Rule $\circ\rhd_R$ is sound iff $R_\Box \subseteq R_\Diamond$.*

Proof. ($\Leftarrow$) We show that if the frame condition holds, then the rule is sound. We assume that: (1) $R_\Box \subseteq R_\Diamond$, and (2) that the formula translation $\Diamond(p \prec \bot) \to \Box(\top \to q)$ of the premise is valid. We then show that the formula translation $\Box(p \to q)$ of the conclusion is valid.

For a contradiction, suppose that $\Box(p \to q)$ is not valid. That is, assume there exists a world u such that $u \not\Vdash \Box(p \to q)$. Then (4) there exist worlds x and y such that $u \leq x$ & $xR_\Box y$ and $y \not\Vdash p \to q$. Thus there exists a world $z \geq y$ and $z \Vdash p$ and $z \not\Vdash q$. Immediately, we also have $z \Vdash p \prec \bot$ and $z \not\Vdash \top \to q$.

The pattern $xR_\Box y \leq z$ implies there is a world w with $x \leq wR_\Box z$ by F2$\Box$. The frame condition (1) then gives $wR_\Diamond z$ too, meaning that $w \Vdash \Diamond(p \prec \bot)$. From (2) we get $w \Vdash \Box(\top \to q)$, which gives us $z \Vdash \top \to q$, giving us the contradiction we seek. Therefore the premise $\Box(p \to q)$ is valid and the rule is sound.

($\Rightarrow$) We show that if the rule is sound, then the failure of the frame condition gives a contradiction. So suppose that the rule is sound. The soundness of the rule implies that $(\Diamond(p \prec \bot) \to \Box(\top \to q)) \to (\Box(p \to q))$ is valid. For a contradiction, suppose we have a frame with $R_\Box \not\subseteq R_\Diamond$. That is, (5): there exist x and y such that $xR_\Box y$ but not $xR_\Diamond y$. The following model $\langle W, \leq, R_\Diamond, R_\Box, V\rangle$ satisfies (5), and has

$$u \Vdash \Diamond(p \prec \bot) \to \Box(\top \to q)$$

but

$$u \not\Vdash \Box(p \to q)$$

- $W \quad = \{u, w, x, y, z\}$
- $< \quad = \{(u, x), (x, w), (y, z)\}$
- $\leq$ is the reflexive, transitive closure of $<$
- $R_\Diamond \quad = \emptyset$
- $R_\Box \quad = \{(x, y), (w, z)\}$
- $V(p) \quad = \{z\}, V(q) = \emptyset$

Thus, we have falsified $(\Diamond(p \prec \bot) \to \Box(\top \to q)) \to (\Box(p \to q))$; however, this contradicts the soundness of the rule. Therefore our assumption was incorrect and the frame condition must hold. Q.E.D.

Lemma 7.6.3. *Rule $\bullet\rhd_R$ is sound iff $R_\Diamond \subseteq R_\Box$.*

Proof. $R_\Diamond \subseteq R_\Box$ means $R_\blacksquare \subseteq R_\blacklozenge$; the rest of the proof is analogous to the proof of Lemma 7.6.2. Q.E.D.

Theorem 7.6.4. *If A is derivable in* **LBiKt**$+\circ\triangleright_R + \bullet\triangleright_R$ *then A is E-valid.*

Proof. Straightforward from the soundness of **LBiKt** w.r.t. `BiKt`-semantics (which subsumes Ewald's semantics) and Lemma 7.6.2 and Lemma 7.6.3. Q.E.D.

Completeness of the extended **LBiKt** w.r.t. IKt and IK can be shown by deriving the axioms of IKt and IK.

Theorem 7.6.5. *System* **LBiKt** $+ \bullet\triangleright_R + \circ\triangleright_R$ *is complete with respect to Ewald's IKt and Simpson's IK.*

Proof. We show the non-trivial cases; the rest are analogous or easier. The following is a derivation of Simpson's axiom 2 and Ewald's axiom 5:

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{id}
\UnaryInfC{$A \Rightarrow A$}
\RightLabel{$\triangleright_L$}
\UnaryInfC{$(A \triangleright \emptyset) \Rightarrow A$}
\RightLabel{w_R}
\UnaryInfC{$(A \triangleright \emptyset) \Rightarrow A, \bullet(\emptyset \triangleright \Diamond B)$}
\AxiomC{}
\RightLabel{id}
\UnaryInfC{$B \Rightarrow B$}
\RightLabel{$\triangleright_L$}
\UnaryInfC{$B \triangleright \emptyset \Rightarrow B$}
\RightLabel{$\Diamond_R$}
\UnaryInfC{$\circ(B \triangleright \emptyset) \Rightarrow \Diamond B$}
\RightLabel{$rp_\bullet$}
\UnaryInfC{$B \Rightarrow \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{w_L}
\UnaryInfC{$B, (A \triangleright \emptyset) \Rightarrow \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{$\to_L$}
\BinaryInfC{$A \to B, (A \triangleright \emptyset) \Rightarrow \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{$\triangleright_R$}
\UnaryInfC{$A \to B \Rightarrow (A \triangleright \emptyset) \triangleright \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{$\Box_L$}
\UnaryInfC{$\Box(A \to B) \Rightarrow \circ((A \triangleright \emptyset) \triangleright \bullet(\emptyset \triangleright \Diamond B))$}
\RightLabel{$\triangleright_L$}
\UnaryInfC{$(\Box(A \to B) \triangleright \emptyset) \Rightarrow \circ((A \triangleright \emptyset) \triangleright \bullet(\emptyset \triangleright \Diamond B))$}
\RightLabel{$rp'_\circ$}
\UnaryInfC{$\bullet(\Box(A \to B) \triangleright \emptyset) \Rightarrow (A \triangleright \emptyset) \triangleright \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{s_R}
\UnaryInfC{$(A \triangleright \emptyset), \bullet(\Box(A \to B) \triangleright \emptyset) \Rightarrow \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{$\triangleright_R$}
\UnaryInfC{$A \triangleright \emptyset \Rightarrow \bullet(\Box(A \to B) \triangleright \emptyset) \triangleright \bullet(\emptyset \triangleright \Diamond B)$}
\RightLabel{$\bullet\triangleright_R$}
\UnaryInfC{$A \triangleright \emptyset \Rightarrow \bullet(\Box(A \to B) \triangleright \Diamond B)$}
\RightLabel{$rp'_\bullet$}
\UnaryInfC{$\circ(A \triangleright \emptyset) \Rightarrow (\Box(A \to B) \triangleright \Diamond B)$}
\RightLabel{$\Diamond_L$}
\UnaryInfC{$\Diamond A \Rightarrow (\Box(A \to B) \triangleright \Diamond B)$}
\RightLabel{s_R}
\UnaryInfC{$\Box(A \to B), \Diamond A \Rightarrow \Diamond B$}
\RightLabel{$\to_R \times 2$}
\UnaryInfC{$\Rightarrow \Box(A \to B) \to (\Diamond A \to \Diamond B)$}
\end{prooftree}
$$

The following is a derivation of Ewald's axiom 7:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \Rightarrow A}\,id}{(A \rhd \emptyset) \Rightarrow A}\,\rhd_L}{(A \rhd \emptyset) \Rightarrow A, \bullet(\emptyset \rhd \bot)}\,w_R \qquad \dfrac{}{\bot, (A \rhd \emptyset) \Rightarrow \bullet(\emptyset \rhd \bot)}\,\bot_L}{A \to \bot, (A \rhd \emptyset) \Rightarrow \bullet(\emptyset \rhd \bot)}\,\to_L}{A \to \bot \Rightarrow (A \rhd \emptyset) \rhd \bullet(\emptyset \rhd \bot)}\,\rhd_R}{\Box(A \to \bot) \Rightarrow \circ((A \rhd \emptyset) \rhd \bullet(\emptyset \rhd \bot))}\,\Box_L}{\Box(A \to \bot) \rhd \emptyset \Rightarrow \circ((A \rhd \emptyset) \rhd \bullet(\emptyset \rhd \bot))}\,\rhd_L}{\bullet(\Box(A \to \bot) \rhd \emptyset) \Rightarrow (A \rhd \emptyset) \rhd \bullet(\emptyset \rhd \bot)}\,rp'_\circ}{(A \rhd \emptyset), \bullet(\Box(A \to \bot) \rhd \emptyset) \Rightarrow \bullet(\emptyset \rhd \bot)}\,s_R}{A \rhd \emptyset \Rightarrow \bullet(\Box(A \to \bot) \rhd \emptyset) \rhd \bullet(\emptyset \rhd \bot)}\,\rhd_R}{A \rhd \emptyset \Rightarrow \bullet(\Box(A \to \bot) \rhd \bot)}\,\bullet\rhd_R}{\circ(A \rhd \emptyset) \Rightarrow \Box(A \to \bot) \rhd \bot}\,rp'_\bullet}{\Diamond A \Rightarrow \Box(A \to \bot) \rhd \bot}\,\Diamond_L}{\Box(A \to \bot), \Diamond A \Rightarrow \bot}\,s_R}{\Rightarrow \Box(A \to \bot) \to (\Diamond A \to \bot)}\,\to_R \times 2$$

Q.E.D.

7.6.3 Regaining classical tense logic `Kt`

To collapse `BiKt` to classical tense logic we add the rules $\bullet\rhd_R$ and $\circ\rhd_L$, giving Ewald's IKt with $R_\Diamond = R_\Box$ via Lemmas 7.6.2-7.6.3, and then add following two rules:

$$\dfrac{X_1, X_2 \Rightarrow Y_1, Y_2}{(X_1 \rhd Y_1), X_2 \Rightarrow Y_2}\,s_L^{-1} \qquad \dfrac{X_1, X_2 \Rightarrow Y_1, Y_2}{X_1 \Rightarrow Y_1, (X_2 \rhd Y_2)}\,s_R^{-1}$$

We can now derive the law of the excluded middle and the law of (dual-)contradiction as shown below:

$$\dfrac{\dfrac{\dfrac{\dfrac{p \Rightarrow p, \bot}{(\emptyset \rhd p), p \Rightarrow \bot}\,s_L^{-1}}{(\emptyset \rhd p) \Rightarrow (p \to \bot)}\,\to_R}{\Rightarrow p, (p \to \bot)}\,s_L}{\Rightarrow p \vee (p \to \bot)}\,\vee_R \qquad \dfrac{\dfrac{\dfrac{\dfrac{p, \top \Rightarrow p}{\top \Rightarrow p, (p \rhd \emptyset)}\,s_R^{-1}}{(\top \prec p) \Rightarrow (p \rhd \emptyset)}\,\prec_L}{p, (\top \prec p) \Rightarrow}\,s_R}{p \wedge (\top \prec p) \Rightarrow}\,\wedge_L$$

7.6.4 Axioms T, 4 and B

By utilising the power of deep inference for tense logic from our previous work in Chapter 6, we can enforce the axioms T, 4, and B by adding numerous propagation rules to **DBiKt**. Below we show the case for B but we need many other nesting combinations for full completeness:

$$\frac{\Sigma^-[A, \circ(\Box A, X \rhd \emptyset)]}{\Sigma^-[\circ(\Box A, X \rhd \emptyset)]}\,B\Box_L \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{p, \circ(\Box p \rhd \emptyset) \rhd p}\,id}{\circ(\Box p \rhd \emptyset) \rhd p}\,B\Box_L}{\Diamond\Box p \rhd p}\,\Diamond_L}{\rhd \Diamond\Box p \to p}\,\to_R$$

We show the cases for T and 4 below:

$$\frac{\Sigma^-[A, \Box A]}{\Sigma^-[\Box A]}\,T\Box \qquad \dfrac{\dfrac{\dfrac{}{p, \Box p \rhd p}\,id}{\Box p \rhd p}\,T\Box}{\rhd \Box p \to p}\,\to_R$$

$$\frac{\Sigma[\Box A, X \rhd \circ(\Box A, Y \rhd Z), W]}{\Sigma[\Box A, X \rhd \circ(Y \rhd Z), W]}\,4\Box_L \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\Box p \rhd \circ(\Box p \rhd \Box p)}\,id}{\Box p \rhd \circ(\emptyset \rhd \Box p)}\,4\Box_L}{\Box p \rhd \Box\Box p}\,\Box_R}{\rhd \Box p \to \Box\Box p}\,\to_R$$

Dual rules allow derivations of $p \to \Diamond p$ and $\Diamond\Diamond p \to \Diamond p$.

Chapter 8

Related work

In this chapter, we survey existing sequent calculi for bi-intuitionistic logic and tense logics, and some extended sequent calculi mechanisms that could potentially be applied to these logics.

8.1 Non cut-free sequent calculi

Although Rauszer presented a sequent calculus for bi-intuitionistic logic [99] and "proved" it cut-free, Pinto and Uustalu have recently given a counter-example [95] to her cut-elimination theorem: the formula $p \to (q \vee (r \to ((p \mathord{-\!\!<} q) \wedge r))$ is valid in bi-intuitionistic logic, but cannot be derived in Rauszer's calculus without the cut rule. Similarly, Pinto and Uustalu's counterexample shows that Crolard's sequent calculus [27] for bi-intuitionistic logic is not cut-free. Pinto and Uustalu's counterexample fails in both Rauszer's and Crolard's calculi because they limit certain sequent rules to singleton succedents or antecedents in the conclusion, and the rules do not capture the interaction between implication and exclusion in a cut-free manner.

8.2 Deep inference

The concept of deep inference in nested sequent calculi has been developed independently by several authors over the last 40 years: Mints [87; 88], Dunn [35], Kashima [73] and more recently Brünnler [17; 19; 20] and Poggiolesi [97].

Note that deep inference in the calculus of structures is a slightly different concept and was pioneered by Guglielmi [60]. In his work, inference rules can be applied deep inside formulae, not just deep inside nested sequent structures. There has also been work on deep inference in the calculus of structures for intuitionistic logic [110]. Moreover, computational linguists use a certain kind of deep inference to apply sequent rules deep inside the antecedents of their sequents, which are trees of formulae due to the non-associativity of comma in the Lambek calculus [3; 13].

We now survey three very similar extended sequent mechanisms, which all use nested sequent calculi and deep inference.

8.2.1 Kashima's calculi

Kashima [73] proposes two kinds of nested sequent calculi for tense logics and some extensions: a shallow inference calculus SKt where rules are only applicable to top-level sequents, and a deep inference sequent calculus S2Kt where rules are applicable at any level. Kashima's first system SKt contains structural connectives (proxies) for $\Diamond$ and $\blacklozenge$, and explicit "turn" rules to capture the interactions between them. Thus, it is similar to display calculi for tense logic [75; 51]. However, unlike in the case of display calculi for tense/modal logics, Kashima does not prove cut-admissibility for this system SKt. Instead, he:

1. shows that SKt is sound with respect to the Kripke semantics for tense logic;
2. gives another calculus S2Kt which allows rules to be applied at arbitrary depth. This calculus has "enter" and "exit" rules which allow formuale to be moved between nodes in the nested sequent tree, similar to our propagation rules of **DKt**;
3. shows that if a sequent has a cut-free proof in S2Kt then it has a cut-free proof in SKt;
4. shows that S2Kt minus cut is complete w.r.t. the Kripke semantics of tense logic.

All of the above together imply the completeness of SKt minus cut. Thus, Kashima achieves a cut-free deep inference nested sequent calculus S2Kt for tense logic. However, unlike our work in Chapter 6, he arrives at S2Kt using a mix of syntactic and semantic methods. In particular, he does not prove syntactic cut-elimination for SKt, nor does he show that S2Kt is syntactically complete w.r.t. SKt.

8.2.2 Brünnler's work on nested sequent calculi

Brünnler [17; 19; 20] and Brünnler and Straßburger [21] have developed two kinds of deep inference nested sequent calculi for the basic modal logic K and several extensions. Both kinds of calculi share the same rule for introducing $\Box$-formulae, and differ by the additional rules for capturing frame axioms such as reflexivity and transitivity. In [17] and [19], frame axioms are captured using different $\Diamond$-propagation rules, while in [21] frame axioms are captured using different structural rules. Brünnler's habilitation thesis [20] covers both approaches.

The benefit of the propagation rule approach is that the calculi obtained this way have semantic completeness proofs and terminating proof search procedures. However, note that Brünnler obtains termination by using essentially a global loop check. More specifically, his proof search procedure blocks the application of rules to "cyclic" nodes in the tree of nested sequents, where a leaf of a sequent is defined as cyclic if there is an inner node in the sequent that carries the same set of formulas [20].

On the other hand, the structural rule approach yields modular sequent calculi, meaning that each combination of these rules gives a calculus that is sound and complete for the corresponding modal logic. Brünnler and Straßburger show that both approaches are inter-derivable: [19] shows that the structural rules are admissible given the propagation rules, while [21] shows that the propagation rules are admissible given the structural rules.

Both kinds of calculi enjoy cut-elimination, although contraction needs special care. In particular, [19] eliminates a multicut rule, even though many logical rules have built-in formula contractions and structural contraction is admissible. On the other hand, structural contraction is not admissible in [21]. Instead, structural contraction is replaced with formula contraction plus some logical rules with built-in contraction and an additional structural rule called "medial"; then a multicut rule is eliminated. The medial rule is essentially a distribution principle which allows to derive $\Sigma[\circ\{\Gamma, \Delta\}]$ from $\Sigma[\circ\{\Gamma\}, \circ\{\Delta\}]$, where $\circ$ is the structural proxy for $\Box$. Recall that in our deep inference nested sequent calculus for tense logic **DKt**, we show instead that this distribution principle is admissible (see Lemmas 6.2.5 and 6.2.6).

8.2.3 Poggiolesi's tree hypersequents

Poggiolesi [97] has developed deep inference nested sequent calculi for the basic modal logic `K` and several extensions; she uses the term "tree hypersequents" and a slightly different syntax than Brünnler. However, the more important differences relate to Poggiolesi's cut rule, cut-elimination procedure and her treatment of contraction.

Poggiolesi's cut rule is multiplicative rather than additive. This approach is more in line with traditional proof theory, however, it presents significant challenges when nested sequents are considered. Poggiolesi defines a product relation on two sequents as a tree merge operation, and then eliminates cut using a "merge" rule similar to the "medial" rule in Brünnler and Straßburger's work [21]. Poggiolesi has admissible formula contraction, but her structural contraction rule is not admissible: it can be defined in terms of formula contraction and the "merge" rule.

8.3 Hypersequents and other extended sequent calculi

Avron's hypersequents have been used for many modal [5], intuitionistic and intermediate logics [7; 25]. Similarly, Dosen's "higher level" sequents [33] can cater for many different logics in one (cut-free) setting: for example, both intuitinistic logic, classical logic and modal logics `S4` and `S5`. But we know of no actual work which uses either framework for tense or intuitionistic logics with a "converse" modality/connective like `Kt` or `BiInt`.

Other examples of extended sequent calculi include Trzesicki's calculus for tense logic [113] and Indrzejczak's multiple sequent calculus [72].

Trzesicki's calculus is an extension of the traditional Gentzen calculus whereby some formulae in the sequent may be tagged, and the sequent itself may be either untagged, or tagged with an $\Box$ or $\blacksquare$ tag. Tagged formulae are in a sense "hidden" in the current sequent, but may be untagged (brought to the surface) by applications of certain rules to tagged sequents. Thus Trzesicki's mechanism has some similarities to display calculi and shallow inference nested sequent calculi, however, his tagging of formulae is not as expressive as the structures used in nested sequent calculi or display calculi. Similarly to display logic, Trzesicki's calculus involves a large degree of non-determinism, which is problematic for proof search: he remarks that "it is not always immediately visible which rule was used as the last in the proof of a formula" [113].

Indrzejczak's multiple sequent calculus for tense logic is another extension of a traditional Gentzen calculus, where the left and right hand side of the sequent is indexed by a number. This indexing system is effectively a syntactic representation of the underlying Kripke model [72], and as such has similarities to labelled sequent calculi [89]. Indrzejczak's calculus lacks a natural notion of a cut rule and cut-elimination: his cut rule is only defined for traditional sequents where both premises and the conclusion are indexed by 0.

8.4 Labelled sequent calculi

Pinto and Uustalu have given a cut-free labelled sequent-calculus for bi-intuitionistic logic [95]. Their calculus uses labelled formulae, thereby utilising some semantic aspects, such as explicit worlds and accessibility, directly in the rules. It is also possible to give proof calculi for many modal and tense logics using semantic methods such as labelled sequent calculi [89] and graph calculi [23; 62], but in this thesis we focus on purely syntactic methods.

We restrict our attention to purely syntactic methods for three reasons:

1. There is a certain proof-theoretic tradition of keeping proof calculi independent of semantics: see, for example, Avron's discussion on "good" proof systems [5].

2. We want to answer the question of whether purely syntactic methods allow us to achieve cut-free completeness for bi-intuitionistic and tense logic, and if so, what extended sequent mechanisms are required to do so.

3. A purely syntactic sequent calculus opens the door for further investigations into the computational content of bi-intuitionistic logic, and brings us one step closer to finding a Curry-Howard isomorphism for it.

However, Pinto and Uustalu's recent work [96] on relating their labelled sequent calculus [95] to our nested sequent calculus $\mathbf{LBiInt_1}$ for bi-intuitionistic logic shows that the two methods are not entirely different. Specifically, [96] gives translations between derivations in the two calculi, thus illustrating that nested sequent calculi and labelled sequent calculi are somewhat related. Of course, this is not entirely surprising, since nested sequent calculi can be seen just as syntactic encodings of Kripke trees.

8.5 Tableaux methods for description logics with inverse roles

Description logics are a family of knowledge representation languages [6] with applications in the semantic web [66]. These logics are syntactic variants of multi-modal logics. In recent years, there have been significant advances in efficient reasoning in very expresive description logics.

For example, Horrocks et al. [69] have developed an efficient decision procedure for $ALCI_{R^+}$, a description logic with transitive and inverse roles. Since $ALCI_{R^+}$ is a syntactic variant of the multi-modal version of tense logic `Kt.S4`, it is also closely related to bi-intuitionistic logic. Indeed, we could obtain a decision procedure for bi-intuitionistic logic by translating it into the tense logic `Kt.S4` [121], and using Horrocks' work. However, a translation into `Kt.S4` provides no insights into bi-intuitionistic logic itself, as it compiles away the constructive aspects of bi-intuitionistic logic.

8.6 Bellin's work on logic for pragmatics

As part of their research in logic for pragmatics, Bellin and his colleagues have studied term calculi for dual intuitionistic logic [10] and proof calculi for logics they call "bi-intuitionistic logic" and "polarized bi-intuitionistic logic" [11; 9]. However, neither their "bi-intuitionistic logic" nor "polarized bi-intuitionistic logic" corresponds to Rauszer's bi-intuitionistic logic, since Bellin's logics do not have any interaction between the `Int` and `DualInt` parts of the logic. Bellin's logics can be embedded into `S4` and bi-modal `S4` respectively, rather than `Kt.S4` as is the case for Rauszer's bi-intuitionistic logic.

Chapter 9

Further work and conclusions

In this chapter, we first outline some directions for further work. We then conclude the thesis by summarising our methodological and technical contributions.

9.1 Further work

9.1.1 Proof search for reflexive-transitive tense logic `Kt.S4`

Recall that we gave a deep inference nested sequent calculus **DS4** for `Kt.S4` in Section 6.3.1. While **DS4** calculus is free of contraction and residuation rules, it is still not immediately suitable for proof search. The problem occurs even for the modal fragment `S4` and has been solved using history-based loop checks [63] as well as dynamic blocking in a labelled tableaux calculus for description logic [69]. Naive backward proof search in `S4` (and hence also `Kt.S4`) can loop due to the interaction of $\Diamond$ and $\Box$ connectives (and similarly, $\blacklozenge$ and $\blacksquare$). The following **DS4** derivation fragment illustrates the problem:

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Diamond\Box A, \Box A, \circ\{A, \Diamond\Box A, \Box A, \circ\{A\}\}\end{array}}{\Diamond\Box A, \Box A, \circ\{A, \Diamond\Box A, \Box A\}}\,\Box}{\Diamond\Box A, \Box A, \circ\{A, \Diamond\Box A\}}\,T_b}{\Diamond\Box A, \Box A, \circ\{A\}}\,4_c}{\Diamond\Box A, \Box A}\,\Box}{\Diamond\Box A}\,T_b
$$

As can be seen above, we can keep applying the rules T_b, $\Box$ and 4_c backwards ad infinitum, creating a infinitely deep nested structure, where each node contains the formulae $A, \Diamond\Box A, \Box A$. We now briefly comment on several potential solutions to this problem.

Dynamic blocking Essentially the same problem we described above has already been solved in a labelled tableaux calculus for the description logic $ALCI_{R^+}$ [69], which is a multimodal variant of tense logic with transitive frames. Their solution uses a technique called *dynamic blocking*. When deciding whether an expansion rule is applicable to a particular node in a tableau, they examine every node on the relevant branch of the tableau, and check whether some other node already contains the relevant formula. A key feature of their approach is that a particular node may be blocked and later become unblocked, as new formulae are propagated to this node. Unfortunately, it is not immediately obvious how to adapt this technique to a purely syntactic framework, unless we impose some side conditions on our rules that refer to whole branches of the nested sequent tree.

Histories Heuerding has given a terminating sequent calculus for S4, which uses *histories* for loop-checking [63]. Histories are sets of formulae stored in each sequent that keep relevant information about sequents previously encountered in backward proof search; the $\Box$ rule is then blocked if the histories already contain the relevant $\Box$-formula. Rather than storing the entire branch of nodes as in the case of dynamic blocking, Heuerding's approach only stores certain $\Box$− and $\Diamond$-formulae that are needed for loop-checking. Heuerding's approach is very elegant, since he can syntactically prove its completeness w.r.t. a sound and complete, but non-terminating history-free calculus for S4.

We conjecture that it is possible to extend Heuerding's approach to Kt.S4, by augmenting the histories with additional sets of formulae that also record the $\blacksquare$− and $\blacklozenge$-formulae that are needed for loop-checking. We made some investigations in this direction, but found it difficult to extend Heuerding's syntactic completeness proof to the tense logic case, when used in a nested sequent calculus rather than a traditional sequent calculus.

Global loop-check Brünnler obtains termination for his nested sequent calculi for modal logics by using essentially a global loop check. More specifically, his proof search procedure blocks the application of rules to "cyclic" nodes in the tree of nested sequents, where a leaf of a sequent is defined as cyclic if there is an inner node in the sequent that carries the same set of formulas [20].

We conjecture that it is possible to extend Brünnler's approach to tense logics, since it is the most general of all the approaches we have outlined in this section, and has been designed for nested sequent calculi. We made some initial investigations in this direction and again found it difficult to prove purely syntactically that such an approach is complete. This is perhaps not surprising, given that Brünnler himself proves the completeness of his approach semantically.

9.1.2 Proof search for display logic

We have only scratched the surface with our work on taming proof search for display logic. It remains to be seen whether deep inference can be used to tame other display calculi with more complex binary residuation principles like those in substructural logics [3]. Our ultimate goal is to obtain a systematic way to "sequentialize" a given display calculus to one with nested sequents, and derive a proof search strategy for the latter. A promising first step in this direction could be to extend our results to the primitive extensions of modal tense logic in a more systematic way than was done in Chapter 6.

9.1.3 First-order bi-logics

In this thesis, we have only considered propositional bi-logics. An interesting question is how many of our techniques can easily be applied in a first-order setting. An approach to this might be to consider quantifiers as modal operators, with appropriate display postulates, such as the ones developed in [118].

9.1.4 Global assumptions

In this thesis, we have focused on the derivability of sequents and the corresponding semantic notion of validity. However, derivability of a formula from a set of global assumptions, which semantically corresponds to logical consequence, is also a very important problem. To cater for deciding derivability of a formula from a set of assumptions in our nested sequent framework, we would need to extend our notion of nested sequents so that the nested structures also contain the global assumptions. Additionally, we would need to take the global assumptions into account whenever we create new nodes inside the nested structures.

9.1.5 Curry-Howard correspondence for bi-logics

As mentioned in Section 2.2.4, bi-intuitionistic logic has potential applications in the area of type theory. Now that we have developed purely syntactic cut-free sequent calculi for bi-intuitionistic logic, finding a Curry-Howard isomorphism for our calculi is the next obvious step in this direction. Pinto and Uustalu suggest that our nested sequent structures would be suitable for this task [95]. We conjecture that using our nested sequent calculi to further develop Crolard's work on co-routines [28] might be a good starting point.

Wansing has given a correspondence between intuitionistic tense logic and a lambda calculus with set-theoretic operators [119]. His treatment of tense logic may prove useful when we want to develop a Curry-Howard isomorphism for bi-intuitionistic logic, since both of these logics have strong similarities.

9.1.6 Optimised decision procedures

Finally, whilst we have developed very simple proof-of-concept implementations of all the proof search calculi presented in this thesis, there is a lot of scope for optimisation in each of the implementations.

As a first step, it would be interesting to investigate how many of the traditional optimisations for tableaux calculi for classical modal logics [64] are still applicable in the case of intuitionistic logic. Empirical observations suggest that our deep inference nested sequent calculi perform much better than either **GBiInt** or $\mathbf{LBiInt}_2$; we would therefore suggest that **DBiInt** might be a promising basis for an efficient theorem prover for `BiInt`.

Secondly, a technique called global caching has shown promise in efficient reasoning for some description logics [54; 91], including those with inverse roles [59]. This technique allows to significantly reduce the number of nodes to explore during proof search by avoiding the redundant expansion of a node that has already been expanded in another part of the search tree. It should be possible to apply this technique for efficient proof search in bi-intuitionistic logic, however, the persistence and reverse persistence properties may cause some side-effects.

9.2 Conclusions

In this thesis we have studied several bi-logics from the perspectives of proof theory and proof search: in particular, bi-intuitionistic logic, tense logic and some extensions of it, and finally bi-intuitionistic tense logic.

Our work was originally motivated by the open problem of finding a cut-free sequent calculus for bi-intuitionistic logic, which we first solved using a framework of derivations and refutations. We then considered the broader problem of proof search in display logic, and developed nested sequent calculi for all our bi-logics that come in "shallow" and "deep" flavours.

Our *methodological contributions* are:

1. In Part I, we developed a framework of derivations and refutations as first class citizens, and gave sequent calculi rules that allow to combine derivations and refutations in order to achieve cut-free completeness in logics where traditional calculi fail.

2. In Part II, we addressed the problem of taming proof search in display calculi by using bi-logics as a case study. Specifically, we showed that for a range of bi-logics, residuation rules and contraction on structures can be shown admissible using deep inference in nested sequent calculi. Since residuation rules and contraction on structures are two of the biggest sources of non-determinism in backward proof search for display calculi, our work is a significant first step towards proof search in general display calculi. Our work is the first which establishes a direct correspondence between proofs in a display-like calculus (with explicit residuation rules) and proofs in a contraction-free deep-inference calculus (with no explicit residuation rules).

Our *technical contributions* are:

1. In Chapter 3, we gave the first cut-free sequent calculus **GBiInt** for bi-intuitionistic logic that is amenable to proof search. We showed **GBiInt** to be sound and complete with respect to Rauszer's semantics of `BiInt`, and gave a decision procedure based on **GBiInt**.

2. In Chapters 4 and 5, we gave nested sequent calculi **LBiInt** and **DBiInt** for bi-intuitionistic logic. **LBiInt** enjoys cut-elimination, while **DBiInt** can be derived from **LBiInt**, is amenable to proof search and syntactically complete with respect to **DBiInt**.

3. In Chapter 6, we gave a syntactic cut-elimination procedure for Kashima's shallow inference nested sequent calculus **SKt** for tense logic, and a new nested deep sequent calculus **DKt** for tense logic and some of its extensions. **DKt** is the first sequent calculus that is amenable to proof search and shown to be syntactically complete with respect to a calculus that has cut-elimination.

4. In Chapter 7, we gave the first cut-free sequent calculus for bi-intuitionistic tense logic. We gave deep and shallow versions of it: the shallow calculus is derived from display logic and enjoys cut-elimination; the deep calculus is complete with respect to the shallow calculus and is free of contraction and display postulate structural rules. We gave a decision procedure for bi-intuitionistic tense logic based on the deep calculus, and showed that the deep calculus is complete with respect to the semantics of bi-intuitionistic tense logic. We also showed how to capture well-known logics like Ewald's intuitionistic tense logic and Simpson's intuitionistic modal logic.

Appendix A

Display calculi

In this chapter, we review the basics of display calculi. We then describe Goré's display calculus δ`BiInt` for bi-intuitionistic logic, and compare it to our nested sequent calculus $\mathbf{LBiInt_1}$ for bi-intuitionistic logic. Finally, we discuss proof search in display logic.

A.1 Goré's display calculus for bi-intuitionistic logic

Belnap's Display Logic [12] (we prefer the term display calculi) is an extremely general proof-theoretical framework. Display calculi can be seen as extensions of Gentzen's sequent calculi: while the only structural connective in traditional sequent calculi is the comma, which is interpreted as a conjunction on the left and a disjunction on the right, in display calculi there is typically a set of structural connectives associated with each family of logical connectives. For example, Goré's display calculus for bi-intuitionistic logic [53] contains Gentzen's comma, but also two binary structural connectives "$>$" and "$<$". Here $>$ is a structural proxy for $\to$, and $<$ is a structural proxy for $-\!\!<$. Figure A.1 shows the rules of Goré's calculus, and the bottom four rules highlight the relationship between $>$ and $\to$, and $<$ and $-\!\!<$ respectively.

A display calculus obeys the *display property*: any sequent containing a particular formula occurrence A can be transformed into another sequent in which the occurrence of A is either the whole of the antecedent or the whole of the succedent, using only a subset of the rules called the *display postulates*. The occurrence of A is then said to be displayed. The display property of a display calculus relies on *residuation principles*, which are relationships between different structural connectives. For example, Goré's display postulate rules (see Figure A.1) for the connectives $>$ and $<$ implement residuation between comma and $>$ as well as between comma and $<$.

The most pleasing property of display calculi is that if the rules of the display calculus enjoy eight easily checked conditions, then the calculus is guaranteed to obey cut-admissibility [12]. That is, one single cut-admissibility proof suffices for all display calculi. This modularity makes it an excellent framework for designing sequent calculi for logics, particularly when we wish to mix and match the intuitionistic, modal, or substructural[1] aspects of different logics into a new logic. This approach has been used e.g. by Wansing [116] and Goré [51] to develop display calculi frameworks for a range of modal and substructural logics respectively. Indeed, Goré's display calculus δ`BiInt` for bi-intuitionistic logic [53] is a specific instance of his general display framework where a display calculus can be obtained for any substructural logic in a systematic and modular way [51].

We now review a syntactic variant of δ`BiInt` and compare it to our nested sequent calculi we introduced in earlier chapters. Formally, a δ`BiInt` structure is defined by the following grammar, where A is a `BiInt` formula:

$$X := \emptyset \mid A \mid (X, X) \mid X > X \mid X < X.$$

A δ`BiInt` sequent is of the form $X \Rightarrow Y$, where X and Y are δ`BiInt` structures. The set of δ`BiInt` rules is given in Figure A.1.

As can be observed in the rules of δ`BiInt`, the connectives $>$ and $<$ allow us to form structures by nesting them inside one another, and the display postulates allow us to reorganise the structures so as to bring sub-structures to the top level or hide them. For example, the $<_L$ rule in Figure A.1 can be read downwards as: given a structure $Z < Y$ on the LHS, bring Z to the top level (display Z) while forming the structure X, Y on the RHS. The same rule can be read upwards as: given a structure X, Y on the RHS, display X by hiding away Y in $Z < Y$ on the LHS.

We now illustrate the use of δ`BiInt` by showing a derivation of Uustalu's sequent [95] (see also Example 2.2.4).

Example A.1.1. *The following is a cut-free derivation of Uustalu's sequent in* δ`BiInt`*. As before, the axioms contain the atoms p, q and r. When read backwards from the root, the derivation uses the structural rule $>_L$ to "hide" the structure $q > p$ while the rules for implication and conjunction on the right are applied. It then uses $>_L$, $<_L$ and commutativity to transform $q > p$ into $p < q$, so that finally the rule for exclusion on the right can be applied.*

$$
\begin{prooftree}
\AxiomC{$p \Rightarrow p$}
\AxiomC{$q \Rightarrow q$}
\RightLabel{$-\!\!<_R$}
\BinaryInfC{$p < q \Rightarrow p -\!\!< q$}
\RightLabel{$<_L$}
\UnaryInfC{$p \Rightarrow p -\!\!< q, q$}
\RightLabel{com_R}
\UnaryInfC{$p \Rightarrow q, p -\!\!< q$}
\RightLabel{$>_L$}
\UnaryInfC{$q > p \Rightarrow p -\!\!< q$}
\AxiomC{$r \Rightarrow r$}
\RightLabel{$\wedge_R$}
\BinaryInfC{$(q > p), r \Rightarrow (p -\!\!< q) \wedge r$}
\RightLabel{com_L}
\UnaryInfC{$r, (q > p) \Rightarrow (p -\!\!< q) \wedge r$}
\RightLabel{$>_R$}
\UnaryInfC{$q > p \Rightarrow r > ((p -\!\!< q) \wedge r)$}
\RightLabel{$\rightarrow_R$}
\UnaryInfC{$q > p \Rightarrow r \rightarrow ((p -\!\!< q) \wedge r)$}
\RightLabel{$>_L$}
\UnaryInfC{$p \Rightarrow q, r \rightarrow ((p -\!\!< q) \wedge r)$}
\end{prooftree}
$$

[1]Briefly, substructural logics are characterised by the lack of one or more structural rules that are present in the Gentzen calculus for classical logic - see e.g. Restall [101] for details. For example, relevant logic is characterised by lack of weakening [2], and linear logic is characterised by a lack of weakening and contraction [47].

Display postulates and cut:

$$\begin{array}{c} Z < Y \Rightarrow X \\ \hline\hline Z \Rightarrow X, Y \\ \hline\hline X > Z \Rightarrow Y \end{array} \begin{array}{l} <_L \\ >_L \end{array} \qquad \begin{array}{c} X \Rightarrow Z < Y \\ \hline\hline X, Y \Rightarrow Z \\ \hline\hline Y \Rightarrow X > Z \end{array} \begin{array}{l} <_R \\ >_R \end{array} \qquad \frac{X \Rightarrow A \qquad A \Rightarrow Y}{X \Rightarrow Y}\,cut$$

Basic structural rules:

$$\frac{\emptyset \Rightarrow X}{Y \Rightarrow X}\,Ver\emptyset \qquad \frac{X \Rightarrow \emptyset}{X \Rightarrow Y}\,Efq\emptyset \qquad \begin{array}{c} X, \emptyset \Rightarrow Y \\ \hline\hline X \Rightarrow Y \\ \hline\hline \emptyset, X \Rightarrow Y \end{array} \begin{array}{l} \emptyset^{+}_{-L} \\ \emptyset^{+}_{-L} \end{array} \qquad \begin{array}{c} X \Rightarrow \emptyset, Y \\ \hline\hline X \Rightarrow Y \\ \hline\hline X \Rightarrow Y, \emptyset \end{array} \begin{array}{l} \emptyset^{+}_{-R} \\ \emptyset^{+}_{-R} \end{array}$$

Further structural rules:

$$\frac{X \Rightarrow Z}{X, Y \Rightarrow Z}\,gw_L \qquad \frac{Y \Rightarrow Z}{X, Y \Rightarrow Z}\,gw_L \qquad \frac{Z \Rightarrow X}{Z \Rightarrow X, Y}\,gw_R \qquad \frac{Z \Rightarrow Y}{Z \Rightarrow X, Y}\,gw_R$$

$$\frac{X, X \Rightarrow Z}{X \Rightarrow Z}\,gc_L \qquad \frac{Z \Rightarrow X, X}{Z \Rightarrow X}\,gc_R$$

$$\begin{array}{c} X, (Y, Z) \Rightarrow W \\ \hline\hline (X, Y), Z \Rightarrow W \end{array}\,ass_L \qquad \begin{array}{c} W \Rightarrow (X, Y), Z \\ \hline\hline W \Rightarrow X, (Y, Z) \end{array}\,ass_R \qquad \frac{Y, X \Rightarrow Z}{X, Y \Rightarrow Z}\,com_L \qquad \frac{Z \Rightarrow Y, X}{Z \Rightarrow X, Y}\,com_R$$

Logical introduction rules:

$$\bot \Rightarrow \emptyset \;\; \bot_L \qquad \frac{Z \Rightarrow \emptyset}{Z \Rightarrow \bot}\,\bot_R \qquad p \Rightarrow p \;\; id \qquad \frac{\emptyset \Rightarrow Z}{\top \Rightarrow Z}\,\top_L \qquad \emptyset \Rightarrow \top \;\; \top_R$$

$$\frac{A, B \Rightarrow Z}{A \wedge B \Rightarrow Z}\,\wedge_L \qquad \frac{X \Rightarrow A \qquad Y \Rightarrow B}{X, Y \Rightarrow A \wedge B}\,\wedge_R$$

$$\frac{A \Rightarrow X \qquad B \Rightarrow Y}{A \vee B \Rightarrow X, Y}\,\vee_L \qquad \frac{Z \Rightarrow A, B}{Z \Rightarrow A \vee B}\,\vee_R$$

$$\frac{A < B \Rightarrow Z}{A \mathrel{-\!\!<} B \Rightarrow Z}\,\mathrel{-\!\!<}_L \qquad \frac{A \Rightarrow X \qquad Y \Rightarrow B}{X < Y \Rightarrow A \mathrel{-\!\!<} B}\,\mathrel{-\!\!<}_R$$

$$\frac{X \Rightarrow A \qquad B \Rightarrow Y}{A \rightarrow B \Rightarrow X > Y}\,\rightarrow_L \qquad \frac{Z \Rightarrow A > B}{Z \Rightarrow A \rightarrow B}\,\rightarrow_R$$

Figure A.1: A syntactic variant of Goré's display calculus δ`BiInt` for bi-intuitionistic logic [53]. Double lines indicate that the rule may be used both reading from top to bottom and vice versa. Note that Goré's calculus uses a semicolon instead of a comma for the structural proxy for conjunction/disjunction, but we have kept Gentzen's comma in order to make the presentation of display calculi more consistent with other sequent calculi discussed in this thesis. Also, Goré uses $\vdash$ for the sequent turnstile, while we use $\Rightarrow$ and reserve $\vdash$ for meta-level derivability. Finally, we use the connectives $\wedge$ and $\vee$ for conjunction and disjunction, rather than Goré's $\otimes$ and $\oplus$ respectively.

A.1.1 Belnap's conditions

In this section we review Belnap's eight conditions for cut-elimination [12]. As Belnap proves, any display calculus that satisfies these conditions is guaranteed to obey cut-elimination. We start by stating Belnap's definitions of parameter and congruence.

Definition A.1.2 (Parameter)**.** *Given an instance of a sequent rule, the constituents of the rule that remain constant when passing from the premises to the conclusion are called parameters.*

Note that in the case of a traditional Gentzen calculus, the parameters are the side formulae.

Definition A.1.3 (Congruence)**.** *Given an instance of a sequent rule, the constituents occupying similar positions in occurrences of structures assigned to the same structure-variable are called congruent.*

We now describe each of Belnap's conditions, using δ`BiInt` as an illustration.

C1 *Preservation of formulas.* This condition requires that for each rule instance, each formula which is a constituent of some premise is a subformula of some formula in the conclusion. This condition is easily verified by inspection of δ`BiInt` rules. For example, in the $\wedge_R$ rule, the formula B is a constituent of the right premise, and is also a subformula of $A \wedge B$ which occurs in the conclusion.

C2 *Shape-alikeness of parameters.* This condition requires that congruent parameters are occurrences of the same structure, and as Belnap points out [12], it follows immediately from Definition A.1.3 regardless of the sequent rules.

C3 *Non-proliferation of parameters.* This condition requires that each parameter is congruent to at most one constituent in the conclusion of each rule instance. This holds for δ`BiInt` because each structure variable occurs exactly once in the conclusion of each rule. For example, the gc_L rule contains the structure X twice in the premise and once in the conclusion. Note that an inverse of the contraction rule would break this condition.

C4 *Position-alikeness of parameters.* This condition requires that congruent parameters are either all antecedent or all succedent parts of the respective sequents. This condition is also easily verified by inspection of δ`BiInt` rules.

C5 *Display of principal constituents.* This condition requires that for each rule, the principal formula is either the entire antecedent or the entire succedent of the conclusion. This is perhaps one of the most striking differences between display calculi and other sequent calculi formalisms, since this condition demands that there can be no side formulae/structures in the antecedent of left introduction rules, or in the succedent of the right introduction rules. This condition is also easily verified by inspection of δ`BiInt` rules. For example, the $\wedge_L$ rule demands that $A \wedge B$ occupies the entire antecedent of the conclusion.

Note that this condition may seem limiting, however, the display property ensures that any sequent can be transformed into a sequent where the desired substructure is either the whole antecedent or the whole succedent.

C6 *Closure under substitution for succedent parameters.* This condition requires that each rule is closed under simultaneous substitution of arbitrary structures for congruent formulas which are succedent parts. In other words, it requires that all structural rules e.g. contraction are expressed in terms of general structures, rather than just formulae. This is another significant difference between display calculi and traditional sequent calculi, where structural rules are generally expressed in terms of formulae. This condition is also easily verified by inspection of δ`BiInt` rules.

C7 *Closure under substitution for antecedent parameters.* Same as C6, but applies to the antecedent of each rule.

C8 *Eliminability of matching principal constituents.* This condition requires that for every two rule instances with conclusions (1) $X \Rightarrow M$ and (2) $M \Rightarrow Y$ with M principal in both, either:

- (3) $X \Rightarrow Y$ is identical to (1) or (2), or
- it is possible to derive (3) from (1) and (2) using the rules of the calculus plus the following rule, where X' and Y' are arbitrary but M' is restricted to proper subformulas of M:

$$\dfrac{X' \Rightarrow M' \qquad M' \Rightarrow Y'}{X' \Rightarrow Y'}$$

Showing that a calculus obeys C8 effectively amounts to showing how to reduce a cut on some formula to one or more cuts on its subformulae, in the case where the cut formula is principal in both derivations above cut. We need to check the cases for all logical connectives of δ`BiInt`; here we just illustrate the case for implication and refer the reader to [51; 53] for details. Suppose $M = A \to B$ and the given derivation ends as follows:

$$\dfrac{\dfrac{X \Rightarrow A > B}{X \Rightarrow A \to B} \qquad \dfrac{Y \Rightarrow A \qquad B \Rightarrow Z}{A \to B \Rightarrow Y > Z}}{X \Rightarrow Y > Z}\, cut$$

We then obtain the following derivation which uses cuts on the subformulae A and B:

$$\dfrac{\dfrac{\dfrac{Y \Rightarrow A \qquad \dfrac{\dfrac{\dfrac{X \Rightarrow A > B}{A, X \Rightarrow B}\, >_R \qquad B \Rightarrow Z}{A, X \Rightarrow Z}\, cut}{A \Rightarrow Z < X}\, <_R}{Y \Rightarrow Z < X}\, cut}{Y, X \Rightarrow Z}\, <_R}{X \Rightarrow Y > Z}\, >_R$$

A.1.2 Relating Goré's δBiInt to our LBiInt$_1$

The differences between our calculus **LBiInt$_1$** presented in Chapter 4 and Goré's δ`BiInt` fall into two groups: differences in structures (items (1) to (3) below) and differences in logical rules (item (4) below). We now describe these differences in detail.

1. δ`BiInt` has explicit rules for associativity and commutativity of comma (ass_L, ass_R, com_L, com_R), as well as manipulating the empty structure ($Ver\emptyset$, $Efq\emptyset$, $\emptyset^{+}_{-L}$, $\emptyset^{+}_{-R}$). On the other hand, in **LBiInt$_1$** we simply stated that we consider structures modulo associativity of comma and assume the empty structure is unitary. As a result, **LBiInt$_1$** lacks the rules ass_L, ass_R, com_L, com_R, $Ver\emptyset$, $Efq\emptyset$, $\emptyset^{+}_{-L}$, $\emptyset^{+}_{-R}$ that are present in δ`BiInt`.

2. As Goré points out [53], in the presence of commutativity of the comma (semicolon in his notation), we only require one of $<$ and $>$. That is, $Y < X$ may be replaced by $X > Y$ and vice versa without losing provability. Now, notice that other than the display postulates, the only δ`BiInt` rules which create $>$ and $<$ rules (when viewed backwards) are the introduction rules for $\rightarrow$ on the right and $-\!\!<$ on the left. Specifically, these rules manipulate $<$ on the left and $>$ on the right. So in **LBiInt$_1$** we simplified the structures further and replaced both $<$ and $>$ by $\triangleright$. We then interpret $\triangleright$ as a $<$ or $-\!\!<$ in a negative context and we interpret $\triangleright$ as a $>$ or $\rightarrow$ in a positive context - recall Figure 4.1.

3. The **LBiInt$_1$** rules $\triangleright_L$ and $\triangleright_R$ are one half of the display postulates of δ`BiInt`. **LBiInt$_1$** then has additional rules called s_L and s_R, which can be seen as more general versions of the other half of display postulates. These rules can also be derived in δ`BiInt`, as we show shortly in Lemma A.1.4.

4. Every logical rule in δ`BiInt` requires the principal formula to be *displayed*, that is, it must occupy either the entire right hand side or the entire left hand side of the sequent - recall Belnap's C5 above. However, in **LBiInt$_1$** we allow side structures for all our logical rules, in order to bring **LBiInt$_1$** closer to traditional Gentzen systems. The logical rules of **LBiInt$_1$** can all be derived in δ`BiInt`. For example, the following is a δ`BiInt` derivation of the **LBiInt$_1$** rule for conjunction on the right, where we "wrap" a δ`BiInt` $\wedge_R$ rule instance between display/undisplay rules, and we use contraction on structures because disjunction is additive in **LBiInt$_1$** but multiplicative in δ`BiInt`:

$$\dfrac{\dfrac{\dfrac{\dfrac{X \Rightarrow A, Y}{X < Y \Rightarrow A}\,{<_L} \qquad \dfrac{X \Rightarrow B, Y}{X < Y \Rightarrow B}\,{<_L}}{(X < Y), (X < Y) \Rightarrow A \wedge B}\,{\wedge_R}}{(X < Y) \Rightarrow A \wedge B}\,{gc_L}}{X \Rightarrow A \wedge B, Y}\,{<_L}$$

Similar δ`BiInt` derivations may be obtained for other logical rules of **LBiInt$_1$**.

$$\begin{array}{cl}
\vdots & \\
A \Rightarrow B > C & \\ \hline
A \Rightarrow C, (B > C) & gw_R \\ \hline
(A < (B > C)) \Rightarrow C & <_L \\ \hline
B, (A < (B > C)) \Rightarrow C & gw_L \\ \hline
A < (B > C) \Rightarrow B > C & >_R \\ \hline
A \Rightarrow (B > C), (B > C) & <_L \\ \hline
A \Rightarrow B > C & gc_R \\ \hline
A \Rightarrow B \to C & \to_R
\end{array}$$

Figure A.2: Non-terminating backward proof search attempt in δ`BiInt`

Lemma A.1.4. *The following rules can be derived* δ`BiInt`*:*

$$\frac{(X_1 < Y_1), X_2 \Rightarrow Y_2}{X_1, X_2 \Rightarrow Y_1, Y_2}\, s_L \qquad\qquad \frac{X_1 \Rightarrow Y_1, (X_2 > Y_2)}{X_1, X_2 \Rightarrow Y_1, Y_2}\, s_R$$

Proof. We show the case for s_L; the case for s_R is symmetric.

$$\begin{array}{cl}
(X_1 < Y_1), X_2 \Rightarrow Y_2 & \\ \hline
X_2 \Rightarrow (X_1 < Y_1) > Y_2 & >_R \\ \hline
X_2 \Rightarrow ((X_1 < Y_1) > Y_2), Y_1 & gw_R \\ \hline
X_1, X_2 \Rightarrow ((X_1 < Y_1) > Y_2), Y_1 & gw_L \\ \hline
(X_1, X_2) < Y_1 \Rightarrow (X_1 < Y_1) > Y_2 & <_L \\ \hline
(X_1 < Y_1), ((X_1, X_2) < Y_1) \Rightarrow Y_2 & >_R \\ \hline
X_1 < Y_1 \Rightarrow Y_2 < ((X_1, X_2) < Y_1) & <_R \\ \hline
X_1 \Rightarrow (Y_2 < ((X_1, X_2) < Y_1)), Y_1 & <_L \\ \hline
X_1, X_2 \Rightarrow (Y_2 < ((X_1, X_2) < Y_1)), Y_1 & gw_L \\ \hline
(X_1, X_2) < Y_1 \Rightarrow Y_2 < ((X_1, X_2) < Y_1) & <_L \\ \hline
((X_1, X_2) < Y_1), ((X_1, X_2) < Y_1) \Rightarrow Y_2 & <_R \\ \hline
(X_1, X_2) < Y_1 \Rightarrow Y_2 & gc_L \\ \hline
X_1, X_2 \Rightarrow Y_2, Y_1 & <_L \\ \hline
X_1, X_2 \Rightarrow Y_1, Y_2 & com_R
\end{array}$$

Q.E.D.

A.2 Proof search in display logic

While display calculi are very powerful because of their generality, they have a number of disadvantages for backward proof search. Firstly, the display postulates can and must create large structures during the process of displaying a particular formula occurrence. More specifically, the invertible display postulate rules (for example, $>_L$, $>_R$, $<_L$ and $<_R$ of δ`BiInt` presented previously) allow "pointless" shuffling of structures and easily lead to non-termination of proof search if applied naively. Since these rules are at the heart of display calculi and guarantee the display property, eliminating them without losing the display property is not obvious.

Another issue is the presence of explicit contraction and weakening rules in display calculi which are couched in terms of structures rather than formulae. Replacing these rules with ones based on formulae can break Belnap's condition (C6/C7) that "each rule is closed under simultaneous substitution of arbitrary structures for congruent formulas" [75]. Absorbing them completely to obtain a "contraction-free" calculus is thus not an obvious step.

Figure A.2 illustrates both problems. Reading this derivation attempt backwards, we start with the formula $B \to C$ in the succedent, which we then transform into the structure $B > C$. We then apply contraction and a number of display postulate rules, as well as weakening, and arrive at the sequent $A \Rightarrow B > C$. Since the latter is the same as the second lowest sequent in the derivation, we could potentially repeat this series of steps (and many others) ad inifinitum.

To sum up, a disciplined proof-theoretic methodology for transforming a display calculus into a more manageable traditional "contraction-free" and "display postulate free" calculus whilst preserving cut-admissiblity is an important goal. Although display calculi were not designed for automated proof-search there is a surprising lack of interest in the study of proof search for display logics: the only exceptions are the works of Wansing [117] and Restall [101].

Appendix B

Additional proofs

B.1 Proofs for Chapter 5

Lemma 5.2.6 Admissibility of s_R

For any context $\Sigma[]$, if$\vdash_{\mathbf{DBiInt}} \Pi : \Sigma[X \triangleright Y, (Z \triangleright W)]$ then $\vdash_{\mathbf{DBiInt}} \Pi' : \Sigma[X, Z \triangleright Y, W]$ such that $|\Pi'| \leq |\Pi|$.

Proof.

- First we show the base case when $\Sigma[] = []$. We use a sub-induction on the height of the derivation Π, and obtain Π'_1 (resp. Π'_2) from Π_1 (resp. Π_2) using the sub-induction hypothesis.

  – Cases when Π ends with a propagation rule that moves formulae within the structures Z and W:

$$\dfrac{\begin{matrix}\Pi_1\\ X \triangleright Y, (A, (A, Z_1 \triangleright Z_2) \triangleright W)\end{matrix}}{X \triangleright Y, ((A, Z_1 \triangleright Z_2) \triangleright W)}\triangleright_{L1} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi'_1\\ X, A, (A, Z_1 \triangleright Z_2) \triangleright Y, W\end{matrix}}{X, (A, Z_1 \triangleright Z_2) \triangleright Y, W}\triangleright_{L1}$$

$$\dfrac{\begin{matrix}\Pi_1\\ X \triangleright Y, (Z \triangleright (W_1 \triangleright W_2, A), A)\end{matrix}}{X \triangleright Y, (Z \triangleright (W_1 \triangleright W_2, A))}\triangleright_{R1} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi'_1\\ X, Z \triangleright Y, (W_1 \triangleright W_2, A), A\end{matrix}}{X, Z \triangleright Y, (W_1 \triangleright W_2, A)}\triangleright_{R1}$$

  – Cases when Π ends with a propagation rule that moves formulae between the structures X, Y, Z, W:

$$\dfrac{\begin{matrix}\Pi_1\\ A, X \triangleright Y, (A, Z \triangleright W)\end{matrix}}{A, X \triangleright Y, (Z \triangleright W)}\triangleright_{L2} \quad \rightsquigarrow \quad \dfrac{\begin{matrix}\Pi'_1\\ A, A, X, Z \triangleright Y, W\end{matrix}}{A, X, Z \triangleright Y, W}\text{ Lemma 5.2.3}$$

$$\dfrac{\overset{\Pi_1'}{X \rhd Y, ((Z_1 \rhd Z_2, A) \rhd W, A)}}{X \rhd Y, ((Z_1 \rhd Z_2) \rhd W, A)} \rhd_{R2} \quad \rightsquigarrow \quad \dfrac{\overset{\Pi_1'}{X, (Z_1 \rhd Z_2, A) \rhd Y, W, A}}{X, (Z_1 \rhd Z_2) \rhd Y, W, A} \rhd_{R2}$$

– Case when Π ends with the logical rule $\to_L$ where the principal formula is in Z:

$$\dfrac{\overset{\Pi_1}{X \rhd Y, (A \to B, Z_1 \rhd A, W)} \qquad \overset{\Pi_2}{X \rhd Y, (A \to B, B, Z_1 \rhd W)}}{X \rhd Y, (A \to B, Z_1 \rhd W)} \to_L \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{X, A \to B, Z_1 \rhd A, Y, W} \qquad \overset{\Pi_2'}{X, A \to B, B, Z_1 \rhd Y, W}}{X, A \to B, Z_1 \rhd Y, W} \to_L$$

– Case when Π ends with the logical rule $-\!\!<_R$ where the principal formula is in W:

$$\dfrac{\overset{\Pi_1}{X \rhd Y, (Z \rhd W_1, A -\!\!< B, A)} \qquad \overset{\Pi_2}{X \rhd Y, (Z, B \rhd W_1, A -\!\!< B)}}{X \rhd Y, (Z \rhd W_1, A -\!\!< B)} -\!\!<_R \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{X, Z \rhd Y, W_1, A -\!\!< B, A} \qquad \overset{\Pi_2'}{X, Z, B \rhd Y, W_1, A -\!\!< B}}{X, Z \rhd Y, W_1, A -\!\!< B} -\!\!<_R$$

– The cases involving other rules follow immediately from the sub-induction hypothesis, since they do not move formulae across $\rhd$-structures.

- For the inductive cases, we have either (1) $\Sigma[] = \Sigma_1[([], U) \rhd V]$ or (2) $\Sigma[] = \Sigma_1[U \rhd (V, [])]$ for some (possibly empty) structures U and V and some context $\Sigma_1[]$.

  We first show case (1) when $\Sigma[] = \Sigma_1[([], U) \rhd V]$. We consider sub-cases when a formula is either propagated out from the context $\Sigma[]$ to U, or from V into the context $\Sigma[]$. In each case below, we obtain Π_1' from Π_1 using the induction hypothesis, applied to the context $\Sigma_1[]$.

  – Case when Π ends with a propagation rule that moves a formula out from $\Sigma[]$ to U:

$$\dfrac{\overset{\Pi_1}{\Sigma_1[((X_1, A \rhd Y, (Z \rhd W)), A, U) \rhd V]}}{\Sigma_1[((X_1, A \rhd Y, (Z \rhd W)), U) \rhd V]} \rhd_{L1} \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{\Sigma_1[((X_1, A, Z \rhd Y, W), A, U) \rhd V]}}{\Sigma_1[((X_1, A, Z \rhd Y, W), U) \rhd V]} \rhd_{L1}$$

  – Case when Π ends with a propagation rule that moves a formula from V into $\Sigma[]$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma_1[((X \triangleright Y, (Z \triangleright W), A), U) \triangleright V_1, A]\end{array}}{\Sigma_1[((X \triangleright Y, (Z \triangleright W)), U) \triangleright V_1, A]}\, \triangleright_{R2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma_1[(X, Z \triangleright Y, W, A), U \triangleright V_1, A]\end{array}}{\Sigma_1[(X, Z \triangleright Y, W), U \triangleright V_1, A]}\, \triangleright_{R2}$$

– The cases when formulae are propagated within the context can be proven identically to the case when $\Sigma[] = []$.

We now show case (2) when $\Sigma[] = \Sigma_1[U \triangleright (V, [])]$. We consider sub-cases when a formula is either propagated out from the context $\Sigma[]$ to V, or from U into the context $\Sigma[]$. In each case below, we obtain Π_1' from Π_1 using the induction hypothesis, applied to the context $\Sigma_1[]$.

– Case when Π ends with a propagation rule that moves a formula from $\Sigma[]$ into V:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma_1[U \triangleright (V, (X \triangleright Y_1, (Z \triangleright W), A), A)]\end{array}}{\Sigma_1[U \triangleright (V, (X \triangleright Y_1, (Z \triangleright W), A))]}\, \triangleright_{R1} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma_1[U \triangleright (V, (X, Z \triangleright Y_1, W, A), A)]\end{array}}{\Sigma_1[U \triangleright (V, (X, Z \triangleright Y_1, W, A))]}\, \triangleright_{R1}$$

– Case when Π ends with a propagation rule that moves a formula from U into $\Sigma[]$:

$$\dfrac{\begin{array}{c}\Pi_1\\ \Sigma_1[U_1, A \triangleright (V, (A, X \triangleright Y, (Z \triangleright W)))]\end{array}}{\Sigma_1[U_1, A \triangleright (V, ((X \triangleright Y, (Z \triangleright W)))]}\, \triangleright_{L2} \quad \rightsquigarrow$$

$$\dfrac{\begin{array}{c}\Pi_1'\\ \Sigma_1[U_1, A \triangleright (V, (A, X, Z \triangleright Y, W))]\end{array}}{\Sigma_1[U_1, A \triangleright (V, (X, Z \triangleright Y, W))]}\, \triangleright_{L2}$$

Q.E.D.

Lemma 5.2.8 Admissibility of $\triangleright_R$

For any context $\Sigma[]$ such that either $\Sigma[] = []$ or $\Sigma[]$ is a positive context, if $\vdash_{\mathbf{DBiInt}}$ $\Pi : X, Y \triangleright Z$ then $\vdash_{\mathbf{DBiInt}} \Pi' : X \triangleright (Y \triangleright Z)$.

Proof.

- First we show the base case when $\Sigma[] = []$. We use a sub-induction on the height of the derivation Π, and obtain Π_1' (resp. Π_2') from Π_1 (resp. Π_2) using the sub-induction hypothesis.

- Cases when Π ends with a propagation rule that moves formulae between the structures Y and Z:

$$\dfrac{\overset{\Pi_1}{X, A, Y_1 \triangleright (A, Z_1 \triangleright Z_2)}}{X, A, Y_1 \triangleright (Z_1 \triangleright Z_2)} \triangleright_{L2} \quad \rightsquigarrow \quad \dfrac{\overset{\Pi_1'}{X \triangleright (A, Y_1 \triangleright (A, Z_1 \triangleright Z_2))}}{X \triangleright (A, Y_1 \triangleright (Z_1 \triangleright Z_2))} \triangleright_{L2}$$

$$\dfrac{\overset{\Pi_1}{X, (Y_1 \triangleright Y_2, A) \triangleright Z_1, A}}{X, (Y_1 \triangleright Y_2) \triangleright Z_1, A} \triangleright_{R2} \quad \rightsquigarrow \quad \dfrac{\overset{\Pi_1'}{X \triangleright ((Y_1 \triangleright Y_2, A) \triangleright Z_1, A)}}{X \triangleright ((Y_1 \triangleright Y_2) \triangleright Z_1, A)} \triangleright_{R2}$$

- Cases when Π ends with a propagation rule that moves formulae between the structures X and Z:

$$\dfrac{\overset{\Pi_1}{A, X_1, Y \triangleright (A, Z_1 \triangleright Z_2)}}{A, X_1, Y \triangleright (Z_1 \triangleright Z_2)} \triangleright_{L2} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{\overset{\Pi_1'}{A, X_1 \triangleright (Y \triangleright (A, Z_1 \triangleright Z_2))}}{A, X_1 \triangleright (A, Y \triangleright (A, Z_1 \triangleright Z_2))} \text{ Lm. 5.2.2}}{A, X_1 \triangleright (A, Y \triangleright (Z_1 \triangleright Z_2))} \triangleright_{L2}}{A, X_1 \triangleright (Y \triangleright (Z_1 \triangleright Z_2))} \triangleright_{L2}$$

$$\dfrac{\overset{\Pi_1}{(X_1 \triangleright X_2, A), Y \triangleright Z_1, A}}{(X_1 \triangleright X_2), Y \triangleright Z_1, A} \triangleright_{R2} \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{\overset{\Pi_1'}{(X_1 \triangleright X_2, A) \triangleright (Y \triangleright Z_1, A)}}{(X_1 \triangleright X_2, A) \triangleright (Y \triangleright Z_1, A), A} \text{ Lm. 5.2.2}}{(X_1 \triangleright X_2) \triangleright (Y \triangleright Z_1, A), A} \triangleright_{R2}}{(X_1 \triangleright X_2) \triangleright (Y \triangleright Z_1, A)} \triangleright_{R1}$$

- Cases when Π ends with a propagation rule that moves formulae within the structures Y and Z:

$$\dfrac{\overset{\Pi_1}{X, A, (A, Y_1 \triangleright Y_2) \triangleright Z}}{X, (A, Y_1 \triangleright Y_2) \triangleright Z} \triangleright_{L1} \quad \rightsquigarrow \quad \dfrac{\overset{\Pi_1'}{X \triangleright (A, (A, Y_1 \triangleright Y_2) \triangleright Z)}}{X \triangleright ((A, Y_1 \triangleright Y_2) \triangleright Z)} \triangleright_{L1}$$

$$\dfrac{\overset{\Pi_1}{X, Y \triangleright (Z_1 \triangleright Z_2, A), A}}{X, Y \triangleright (Z_1 \triangleright Z_2, A)} \triangleright_{R1} \quad \rightsquigarrow \quad \dfrac{\overset{\Pi_1'}{X \triangleright (Y \triangleright ((Z_1 \triangleright Z_2, A), A))}}{X \triangleright (Y \triangleright (Z_1 \triangleright Z_2, A))} \triangleright_{R1}$$

- Case when Π ends with a $\rightarrow_L$ rule where the principal formula is in X:

$$\cfrac{\begin{matrix}\Pi_1\\ X_1, A \to B, Y \rhd A, Z\end{matrix}\qquad \begin{matrix}\Pi_2\\ X_1, A \to B, B, Y \rhd Z\end{matrix}}{X_1, A \to B, Y \rhd Z}\,{\to_L}\;\rightsquigarrow$$

$$\cfrac{\cfrac{\cfrac{\begin{matrix}\Pi'_1\\ X_1 \rhd (A \to B, Y \rhd A, Z)\end{matrix}\qquad \begin{matrix}\Pi'_2\\ X_1 \rhd (A \to B, B, Y \rhd Z)\end{matrix}}{X_1 \rhd (A \to B, Y \rhd Z)}\,{\to_L}}{X_1, A \to B \rhd (A \to B, Y \rhd Z)}\,\text{Lemma 5.2.2}}{X_1, A \to B \rhd (Y \rhd Z)}\,{\rhd_{L2}}$$

– Case when Π ends with a $\to_L$ rule where the principal formula is in Y:

$$\cfrac{\begin{matrix}\Pi_1\\ X, A \to B, Y_1 \rhd A, Z\end{matrix}\qquad \begin{matrix}\Pi_2\\ X, A \to B, B, Y_1 \rhd Z\end{matrix}}{X, A \to B, Y_1 \rhd Z}\,{\to_L}\;\rightsquigarrow$$

$$\cfrac{\begin{matrix}\Pi'_1\\ X \rhd (A \to B, Y_1 \rhd A, Z)\end{matrix}\qquad \begin{matrix}\Pi'_2\\ X \rhd (A \to B, B, Y_1 \rhd Z)\end{matrix}}{X \rhd (A \to B, Y_1 \rhd Z)}\,{\to_L}$$

– Case when Π ends with a $-\!\!<_R$ rule where the principal formula is in Z:

$$\cfrac{\begin{matrix}\Pi_1\\ X, Y \rhd Z_1, A -\!\!< B, A\end{matrix}\qquad \begin{matrix}\Pi_2\\ X, Y, B \rhd Z_1, A -\!\!< B\end{matrix}}{X, Y \rhd Z_1, A -\!\!< B}\,{-\!\!<_R}\;\rightsquigarrow$$

$$\cfrac{\begin{matrix}\Pi'_1\\ X \rhd (Y \rhd Z_1, A -\!\!< B, A)\end{matrix}\qquad \begin{matrix}\Pi'_2\\ X \rhd (Y, B \rhd Z_1, A -\!\!< B)\end{matrix}}{X \rhd (Y \rhd Z_1, A -\!\!< B)}\,{-\!\!<_R}$$

– The cases involving other rules follow immediately from the induction hypothesis, since they do not move formulae across $\rhd$-structures.

- For the inductive case, we have $\Sigma[] = \Sigma_1[U \rhd (V, [])]$ for some (possibly empty) structures U and V and some context $\Sigma_1[]$. We now consider sub-cases when a formula is either propagated out from the context $\Sigma[]$ to V, or from U into the context $\Sigma[]$. In each case below, we obtain Π'_1 from Π_1 using the induction hypothesis, applied to the context $\Sigma_1[]$.

  – Case when Π ends with a propagation rule that moves a formula out from $\Sigma[]$ to V:

$$\cfrac{\begin{matrix}\Pi_1\\ \Sigma_1[U \rhd V, (X, Y \rhd Z_1, A), A]\end{matrix}}{\Sigma_1[U \rhd V, (X, Y \rhd Z_1, A)]}\,{\rhd_{R1}}\;\rightsquigarrow$$

$$\cfrac{\cfrac{\cfrac{\begin{matrix}\Pi'_1\\ \Sigma_1[U \rhd V, (X \rhd (Y \rhd Z_1, A)), A]\end{matrix}}{\Sigma_1[U \rhd V, (X \rhd (Y \rhd Z_1, A), A), A]}\,\text{Lemma 5.2.2}}{\Sigma_1[U \rhd V, (X \rhd (Y \rhd Z_1, A), A)]}\,{\rhd_{R1}}}{\Sigma_1[U \rhd V, (X \rhd (Y \rhd Z_1, A))]}\,{\rhd_{R1}}$$

– Case when Π ends with a propagation rule that moves a formula from U into $\Sigma[]$:

$$\dfrac{\overset{\Pi_1}{\Sigma_1[U_1, A \triangleright V, (A, X, Y \triangleright Z)]}}{\Sigma_1[U_1, A \triangleright V, (X, Y \triangleright Z)]}\,\triangleright_{L2} \rightsquigarrow$$

$$\dfrac{\overset{\Pi_1'}{\Sigma_1[U_1, A \triangleright V, (A, X \triangleright (Y \triangleright Z))]}}{\Sigma_1[U_1, A \triangleright V, (X \triangleright (Y \triangleright Z))]}\,\triangleright_{L2}$$

– The cases when formulae are propagated within the context can be proven identically to the case when $\Sigma[] = []$.

Q.E.D.

Bibliography

[1] G. Amati and F. Pirri. A uniform tableau method for intuitionistic modal logics. I. *Studia Logica*, 53(1):29–60, 1994.

[2] A. R. Anderson and N. D. Belnap, Jr. *Entailment: The Logic of Relevance and Necessity*, volume 1. Princeton University Press, Princeton, New Jersey, 1975.

[3] C. Areces and R. Bernardi. Analyzing the core of categorial grammar. *Journal of Logic, Language, and Information*, 13(2):121–137, 2004.

[4] Z. M. Ariola, H. Herbelin, and A. Sabry. A type-theoretic foundation of delimited continuations. *Higher Order and Symbolic Computation*, 22(3):233–273, Sept. 2009.

[5] A. Avron. The method of hypersequents in the proof theory of propositional non-classical logics. In W. H. et al., editor, *Logic: from foundations to applications. Proc. Logic Coll., Keele, UK, 1993*, pages 1–32. Oxford Univ. Press, New York, 1996.

[6] F. Baader and C. Lutz. Description logic. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic*, pages 757–819. Elsevier, 2007.

[7] M. Baaz, A. Ciabattoni, and C. G. Fermüller. Hypersequent calculi for Gödel logics - a survey. *Journal of Logic and Computation*, 13:1–27, 2003.

[8] D. Batens, C. Mortensen, G. Priest, and J. P. V. Bendegem, editors. *Frontiers of Paraconsistent Logic*. Number 8 in Studies in Logic and Computation. Research Studies Press, 2000.

[9] G. Bellin. Natural deduction and term assignment for co-Heyting algebras in polarized bi-intuitionistic logic. Submitted to the Proceedings of the Natural Deduction Conference, Rio de Janeiro, July 2-6, 2001. Accessed from `http://profs.sci.univr.it/~bellin/nat/nat.pdf` on 5th March 2010.

[10] G. Bellin. A term assignment for dual intuitionistic logic. Intuitionistic Modal Logics and Applications Workshop (IMLA '05), 2005. Accessed from `http://profs.sci.univr.it/~bellin/chicago/chicago.pdf` on 5th March 2010.

[11] G. Bellin and C. Biasi. Towards a logic for pragmatics. assertions and conjectures. *Journal of Logic and Computation*, 14(4):473–506, 2004.

[12] N. D. Belnap, Jr. Display logic. *Journal of Philosophical Logic*, 11(4):375–417, 1982.

[13] R. Bernardi and M. Moortgat. Continuation semantics for symmetric categorial grammar. In D. Leivant and R. J. G. B. de Queiroz, editors, *WoLLIC*, volume 4576 of *Lecture Notes in Computer Science*, pages 53–71. Springer, 2007.

[14] P. Blackburn, J. Benthem, and F. Wolter. *Handbook of Modal Logic, Volume 3 (Studies in Logic and Practical Reasoning)*. Elsevier Science Inc., New York, NY, USA, 2006.

[15] L. Brouwer. *Brouwer's Cambridge lectures on intuitionism*. Cambridge University Press, 1981. Edited by D. van Dalen.

[16] K. Brünnler. Atomic cut elimination for classical logic. In M. Baaz and J. A. Makowsky, editors, *CSL*, volume 2803 of *Lecture Notes in Computer Science*, pages 86–97. Springer, 2003.

[17] K. Brünnler. Deep sequent systems for modal logic. In G. G. et al, editor, *Advances in Modal Logic*, page 107. College Publications, 2006.

[18] K. Brünnler. Deep sequents for modal logic. Unpublished, 2007.

[19] K. Brünnler. Deep sequent systems for modal logic. *Arch. Math. Log.*, 48(6):551–577, 2009.

[20] K. Brünnler. Nested sequents. *CoRR*, abs/1004.1845, 2010.

[21] K. Brünnler and L. Straßburger. Modular sequent systems for modal logic. In M. Giese and A. Waller, editors, *TABLEAUX 09: Automated Reasoning with Analytic Tableaux and Related Methods*, number 5607 in LNAI, pages 152–166, 2009.

[22] L. Buisman (Postniece) and R. Goré. A cut-free sequent calculus for bi-intuitionistic logic. In N. Olivetti, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 4548 of *LNAI*, pages 90–106. Springer, 2007.

[23] M. A. Castilho, L. F. D. Cerro, O. Gasquet, and A. Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3/4):281–297, 1997.

[24] A. Church. A set of postulates for the foundation of logic. *Annals of Mathematics*, 33:346–366, 1932.

[25] A. Ciabattoni and M. Ferrari. Hypersequent calculi for some intermediate logics with bounded Kripke models. *Journal of Logic and Computation*, 11(2):283–294, 2001.

[26] W. Craig. Three uses of the Herbrand-Genzen theorem in relating model theory and proof theory. *Journal of Symbolic Logic*, 22:269–285, 1957.

[27] T. Crolard. Subtractive logic. *Theoretical Computer Science*, 254(1–2):151–185, Mar. 2001.

[28] T. Crolard. A formulae-as-types interpretation of Subtractive Logic. *Journal of Logic and Computation*, 14(4):529–570, August 2004.

[29] P.-L. Curien and H. Herbelin. The duality of computation. In *ICFP '00: Proceedings of the fifth ACM SIGPLAN international conference on Functional programming*, pages 233–243, New York, NY, USA, 2000. ACM Press.

[30] J. Czermak. A remark on Gentzen's calculus of sequents. *Notre Dame Journal of Formal Logic*, 18(3):471–474, 1977.

[31] R. Davies and F. Pfenning. A modal analysis of staged computation. *Journal of the ACM*, 48(3):555–604, May 2001.

[32] E. Donovan. Automated proof search in bi-intuitionistic logic using sequent calculi. Honours thesis, Australian National University, 2005.

[33] K. Dosen. Sequent-systems for modal logic. *The Journal of Symbolic Logic*, 50(1):149–168, 1985.

[34] A. Dragalin. *Mathematical Intuitionism: Introduction to Proof Theory*, volume 68 of *Translations of Mathematical Monographs*. Cambridge Univ. Press, 1988.

[35] M. Dunn. A 'Gentzen' system for positive relevant implication. (abstract). *The Journal of Symbolic Logic*, 38:356–357, 1974.

[36] R. Dyckhoff. Contraction-free sequent calculi for intuitionistic logic. *The Journal of Symbolic Logic*, 57(3):795–807, September 1992.

[37] U. Egly. A polynomial translation of propositional s4 into propositional intuitionistic logic. Unpublished, 1997.

[38] H. B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, New York, 1972.

[39] W. B. Ewald. Intuitionistic tense and modal logic. *The Journal of Symbolic Logic*, 51(1):166–179, 1986.

[40] S. Feferman, editor. *Gödel's Collected Works*. Oxford University Press, Oxford, 1980.

[41] D. Fernandez. A polynomial translation of s4 into intuitionistic logic. *The Journal of Symbolic Logic*, 71:989–1001, 2006.

[42] M. Ferrari. Cut-free tableau calculi for some intuitionistic modal logics. *Studia Logica*, 59(3):303–330, 1997.

[43] A. Filinski. Declarative continuations: an investigation of duality in programming language semantics. In D. H. Pitt, D. E. Rydeheard, P. Dybjer, A. M. Pitts, and A. Poigné, editors, *Category Theory and Computer Science*, volume 389 of *Lecture Notes in Computer Science*, pages 224–249, Berlin, 1989. Springer-Verlag.

[44] M. Fitting. Modal proof theory. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic*, pages 86–138. Elsevier, 2007.

[45] J. H. Gallier. *Logic for Computer Science, Foundations of Automated Theorem Proving*. Computer Science and Technology Series. Harper & Row, 1986.

[46] G. Gentzen. Untersuchungen über das Logische Schliessen. *Mathematische Zeitschrift*, 39:176–210 and 405–431, 1935. English translation in [108].

[47] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[48] K. Gödel. Eine interpretation des intuitionistischen aussagenkalkuls. *Ergebnisse eines Mathematischen Kolloquiums.*, 4:39–40, 1933. English translation in [40].

[49] N. D. Goodman. The logic of contradiction. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 27(2):119–126, 1981.

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

[51] R. Goré. Substructural logics on display. *Logic Journal of the IGPL*, 6(3):451–504, 1998.

[52] R. Goré. Tableau methods for modal and temporal logics. In R. H. Marcello D'Agostino, Dov M. Gabbay and J. Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Kluwer, 1999.

[53] R. Goré. Dual intuitionistic logic revisited. In R. Dyckhoff, editor, *TABLEAUX*, volume 1847 of *Lecture Notes in Computer Science*, pages 252–267. Springer, 2000.

[54] R. Goré and L. A. Nguyen. EXPTIME tableaux with global caching for description logics with transitive roles, inverse roles and role hierarchies. In N. Olivetti, editor, *TABLEAUX*, volume 4548 of *Lecture Notes in Computer Science*, pages 133–148. Springer, 2007.

[55] R. Goré and L. Postniece. Combining derivations and refutations for cut-free completeness in bi-intuitionistic logic. *J. Log. and Comput.*, 20(1):233–260, 2010.

[56] R. Goré, L. Postniece, and A. Tiu. Cut-elimination and proof search for bi-intuitionistic tense logic. In *Advances in Modal Logic 2010*. To appear.

[57] R. Goré, L. Postniece, and A. Tiu. Cut-elimination and proof-search for bi-intuitionistic logic using nested sequents. In C. Areces and R. Goldblatt, editors, *Advances in Modal Logic*, pages 43–66. College Publications, 2008.

[58] R. Goré, L. Postniece, and A. Tiu. Taming displayed tense logics using nested sequents with deep inference. In M. Giese and A. Waaler, editors, *TABLEAUX*, volume 5607 of *Lecture Notes in Computer Science*, pages 189–204. Springer, 2009.

[59] R. Goré and F. Widmann. Sound global state caching for ALC with inverse roles. In M. Giese and A. Waaler, editors, *TABLEAUX*, volume 5607 of *Lecture Notes in Computer Science*, pages 205–219. Springer, 2009.

[60] A. Guglielmi. A system of interaction and structure. *ACM Trans. on Computational Logic*, 8(1):1–64, Jan. 2007.

[61] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004.

[62] A. Heuerding. *Sequent Calculi for Proof Search in some Modal Logics*. PhD thesis, Institute for Applied Mathematics and Computer Science, University of Bern, Switzerland, 1998.

[63] A. Heuerding, M. Seyfried, and H. Zimmermann. Efficient loop-check for backward proof search in some non-classical propositional logics. In *Analytic Tableaux and Related Methods*, volume 1071 of *LNAI*, pages 210–225, 1996.

[64] I. Horrocks. Implementation and optimisation techniques. In F. B. et al., editor, *The Description Logic Handbook: Theory, Implementation, and Applications*, chapter 9, pages 306–346. Cambridge University Press, 2003.

[65] I. Horrocks, U. Hustadt, U. Sattler, and R. Schmidt. Computational modal logic. In P. B. et al., editor, *Handbook of Modal Logic*, pages 181–245. Elsevier, 2007.

[66] I. Horrocks, P. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[67] I. Horrocks and P. F. Patel-Schneider. Optimizing description logic subsumption. *J. Log. Comput*, 9(3):267–293, 1999.

[68] I. Horrocks, U. Sattler, and S. Tobies. A PSpace-algorithm for deciding $ALCNI_{R^+}$-satisfiability. Technical Report LTCS-98-08, LuFG TCS, RWTH Aachen, 1998.

[69] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL*, 8(3):239–264, 2000.

[70] W. A. Howard. The formulas-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press, 1980.

[71] J. M. Howe. *Proof search issues in some non-classical logics*. PhD thesis, University of St Andrews, 1998.

[72] A. Indrzejczak. Multiple sequent calculus for tense logics. International Conference on Temporal Logic, Leipzig 2000, 2000. 93–104.

[73] R. Kashima. Cut-free sequent calculi for some tense logics. *Studia Logica*, 53:119–135, 1994.

[74] D. E. Knuth. *Semantics of Context-Free Languages*, volume 2, pages 127–145. Springer-Verlag, New York, June 1968.

[75] M. Kracht. Power and weakness of the modal display calculus. In H. Wansing, editor, *Proof Theory of Modal Logics*, pages 92–121. Kluwer, 1996.

[76] S. A. Kripke. A completeness theorem in modal logic. *The Journal of Symbolic Logic*, 24, 1959.

[77] S. A. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[78] S. A. Kripke. Semantical analysis of intuitionistic logic. In J. Crossley and M. A. E. Dummett, editors, *Formal Systems and Recursive Functions*, pages 92–130. North-Holland, Amsterdam, 1965.

[79] P. Landin. Correspondence between ALGOL 60 and Church's lambda-notation: part I. *Commun. ACM*, 8(2):89–101, 1965.

[80] D. Larchey-Wendling. Combining proof-search and counter-model construction for deciding Gödel-Dummett logic. In A. Voronkov, editor, *Automated Deduction – CADE-18*, volume 2392 of *LNCS*, pages 94–110, 2002.

[81] J. Łukasiewicz. *Aristotle's syllogistic from the standpoint of modern formal logic*. Clarendon Press, Oxford, 2nd edition, 1957.

[82] P. Łukowski. Modal interpretation of Heyting-Brouwer logic. *Bulletin of the Section of Logic*, 25(2):80–83, 1996.

[83] S. Maehara. Eine darstellung der intuitionistischen logik in der klassischen. *Nagoya Mathematical Journal*, pages 45–64, 1954.

[84] A. Masini. 2-sequent calculus: Intuitionism and natural deduction. *J. Log. Comput*, 3(5):533–562, 1993.

[85] R. P. McArthur. *Tense Logic*. D. Reidel Publishing Co., Dordrecht, Holland, 1976.

[86] K. L. McMillan. Applications of Craig interpolants in model checking. In N. Halbwachs and L. D. Zuck, editors, *TACAS*, volume 3440 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2005.

[87] G. Minc. Cut-elimination theorem in relevant logics. In J. V. Matijasevic and O. A. Silenko, editors, *Isslédovaniá po konstructivnoj mathematiké i matematičeskoj logike V*, pages 90–97. Izdatél'stvo "Nauka", 1972. (English translation in "Cut-Elimination Theorem in Relevant Logics" [88]).

[88] G. Minc. Cut-elimination theorem in relevant logics. *The Journal of Soviet Mathematics*, 6:422–428, 1976. (English translation of the original article [87]).

[89] S. Negri. Proof analysis in modal logic. *Journal of Philosophical Logic*, 34(5):507–544, 2005.

[90] A. Nerode and R. A. Shore. *Logic for applications*. Graduate texts in computer science. Springer, New York, 2nd edition, 1997.

[91] L. A. Nguyen. An efficient tableau prover using global caching for the description logic ALC. *Fundam. Inform*, 93(1-3):273–288, 2009.

[92] F. Pfenning and H.-C. Wong. On a modal λ-calculus for S4. In S. Brookes and M. Main, editors, *Proceedings of the Eleventh Conference on Mathematical Foundations of Programming Semantics*, New Orleans, Louisiana, Mar. 1995. *Electronic Notes in Theoretical Computer Science*, Volume 1, Elsevier.

[93] B. Pierce. *Types and Programming Languages*. The MIT Press, Cambridge, MA, 2002.

[94] L. Pinto and R. Dyckhoff. Loop-free construction of counter-models for intuitionistic propositional logic. In M. B. et al., editor, *Symposia Gaussiana*, pages 225–232, Berlin and New York, 1995. Walter de Gruyter & Co.

[95] L. Pinto and T. Uustalu. Proof search and counter-model construction for bi-intuitionistic propositional logic with labelled sequents. In M. Giese and A. Waaler, editors, *TABLEAUX*, volume 5607 of *Lecture Notes in Computer Science*, pages 295–309. Springer, 2009.

[96] L. Pinto and T. Uustalu. Relating sequent calculi for bi-intuitionistic propositional logic. In S. van Bakel, S. Berardi, and U. Berger, editors, *3rd Workshop on Classical Logic and Computation CL&C 2010*, 2010. To appear.

[97] F. Poggiolesi. The tree-hypersequent method for modal propositional logic. *Trends in Logic: Towards Mathematical Philsophy*, pages 9–30, 2009.

[98] L. Postniece. Deep inference in bi-intuitionistic logic. In H. Ono, M. Kanazawa, and R. J. G. B. de Queiroz, editors, *WoLLIC*, volume 5514 of *Lecture Notes in Computer Science*, pages 320–334. Springer, 2009.

[99] C. Rauszer. A formalization of the propositional calculus of H-B logic. *Studia Logica*, 33:23–34, 1974.

[100] C. Rauszer. An algebraic and Kripke-style approach to a certain extension of intuitionistic logic. *Dissertationes Mathematicae*, 168, 1980. Institute of Mathematics, Polish Academy of Sciences.

[101] G. Restall. *An Introduction to Substructural Logics*. Routledge, London, 2000.

[102] E. Saarinen, editor. *Game-Theoretical Semantics: Essays on Semantics*. D. Reidel Publishing Co., Dordrecht, 1979.

[103] S. Schwendimann. A new one-pass tableau calculus for PLTL. In *Analytic Tableaux and Related Methods*, volume 1397 of *LNAI*, pages 277–292, 1998.

[104] A. K. Simpson. *The proof theory and semantics of intuitionistic modal logic*. PhD thesis, University of Edinburgh, 1994.

[105] E. Spaan. The complexity of propositional tense logics. In M. de Rijke, editor, *Diamonds and Defaults*, pages 287–307. Kluwer Academic Publishers, Dordrecht, 1993.

[106] R. Statman. Intuitionistic propositional logic is polynomial-space complete. *Theoretical Computer Science*, 9(1):67–72, July 1979.

[107] V. Śvejdar. On sequent calculi for intuitionistic propositional logic. *Commentationes Mathematicae Universitatis Carolinae*, 47(1):159–173, 2006.

[108] M. E. Szabo, editor. *The Collected Papers of Gerhard Gentzen*. Studies in Logic and the foundations of Mathematics. North-Holland, Amsterdam, 1969.

[109] S. Thompson. *Haskell: The Craft of Functional Programming (2nd edition)*. Addison Wesley, Reading, Mass., 1999.

[110] A. Tiu. A local system for intuitionistic logic. In M. Hermann and A. Voronkov, editors, *LPAR*, volume 4246 of *Lecture Notes in Computer Science*, pages 242–256. Springer, 2006.

[111] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, 1996.

[112] A. S. Troelstra and D. van Dalen. *Constructivism in mathematics: an introduction*, volume 1. Elsevier Science Publishers, Amsterdam, 1988.

[113] K. Trzesicki. Gentzen-style axiomatization of tense logic. *Bulletin of the Section of Logic*, 13(2):75–83, 1984.

[114] I. Urbas. Dual-intuitionistic logic. *Notre Dame Journal of Formal Logic*, 37(3):440–451, Summer 1996.

[115] Y. Venema. Algebras and co-algebras. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic*, pages 331–426. Elsevier, 2007.

[116] H. Wansing. Sequent calculi for normal modal proposisional logics. *Journal of Logic and Computation*, 4(2):125–142, Apr. 1994.

[117] H. Wansing. Modal tableaux based on residuation. *J. Log. Comput*, 7(6):719–731, 1997.

[118] H. Wansing. *Displaying Modal Logic*. Kluwer Academic Publishers, 1998.

[119] H. Wansing. Formulas-as-types for temporal logic. Technical Report, Dresden University of Technology, Institute of Philosophy, 2000.

[120] H. Wansing. Constructive negation, implication, and co-implication. *Journal of Applied Non-classical Logics*, 18(2–3):341–364, 2008.

[121] F. Wolter. On logics with coimplication. *Journal of Philosophical Logic*, 27(4):353–387, 1998.