Cut-elimination and Proof Search for Bi-Intuitionistic Tense Logic

Rajeev Goré, Linda Postniece and Alwen Tiu

Logic and Computation Group, School of Computer Science
The Australian National University

Abstract

We consider an extension of bi-intuitionistic logic with the traditional modalities ♦, □, • and ■ from tense logic Kt. Proof theoretically, this extension is obtained simply by extending an existing sequent calculus for bi-intuitionistic logic with typical inference rules for the modalities used in display logics. As it turns out, the resulting calculus, LBiKt, seems to be more basic than most intuitionistic tense or modal logics considered in the literature, in particular, those studied by Ewald and Simpson, as it does not assume any a priori relationship between the modal operators ♦ and □. We recover Ewald’s intuitionistic tense logic and Simpson’s intuitionistic modal logic by modularly extending LBiKt with additional structural rules. The calculus LBiKt is formulated in a variant of display calculus, using a form of sequents called nested sequents. Cut elimination is proved for LBiKt, using a technique similar to that used in display calculi. As in display calculi, the inference rules of LBiKt are “shallow” rules, in the sense that they act on top-level formulae in a nested sequent. The calculus LBiKt is ill-suited for backward proof search due to the presence of certain structural rules called “display postulates” and the contraction rules on arbitrary structures. We show that these structural rules can be made redundant in another calculus, DBiKt, which uses deep inference, allowing one to apply inference rules at an arbitrary depth in a nested sequent. We prove the equivalence between LBiKt and DBiKt and outline a proof search strategy for DBiKt. We also give a Kripke semantics and prove that LBiKt is sound with respect to the semantics, but completeness is still an open problem. We then discuss various extensions of LBiKt.

Keywords: Intuitionistic logic, modal logic, intuitionistic modal logic, deep inference.

1 Introduction

Intuitionistic logic Int forms a rigorous foundation for many areas of Computer Science via its constructive interpretation and via the Curry-Howard isomorphism between natural deduction proofs and well-typed terms in the λ-calculus. Central to both concerns are syntactic proof calculi with cut-elimination and backwards proof-search for finding derivations automatically.

In traditional intuitionistic logic, the connectsives → and ∧ form an adjoint pair in that (A ∧ B) → C is valid iff A → (B → C) is valid iff B → (A → C) is valid. Rauszer [22] obtained BiInt by extending Int with a binary connective ¬¬ called “exclusion” which is adjoint to ∨ in that A → (B ∨ C) is valid iff (A ¬¬ B) → C is valid iff (A ¬¬ C) → B is valid. Crolard [4] showed that BiInt has a computational interpretation in terms of continuation passing style semantics. Uustalu and Pinto recently showed that Rauszer’s sequent calculus [21] and Crolard’s extensions of it
fail cut-elimination, but a nested sequent calculus with cut-elimination [9] and a labelled sequent calculus [18] with cut-free-completeness have been found for BiInt.

The literature on Intuitionistic Modal/Tense Logics (IM/TLs) is vast [6,24] and typically uses Hilbert calculi with algebraic, topological or relational semantics. We omit details since our interest is primarily proof-theoretic. Sequent and natural deduction calculi for IMLs are rarer [14,1,17,3,5,12,7]. Extending them with “converse” modalities like ♦ and □ causes cut-elimination to fail as it does for classical modal logic S5 where ♦ is a self-converse. Labels [15,24,16] can help but are not purely proof-theoretic since they encode the Kripke semantics.

The closest to our work is that of Sadrzadeh and Dyckhoff [23] who give a cut-free sequent calculus using deep inference for a logic with an adjoint pair of modalities (♦, □) plus only ∧, ∨, ⊤ and ⊥. As all their connectives are “monotonic”, cut-elimination presents no difficulties.

Let BiKt be the bi-intuitionistic tense logic obtained by extending BiInt with two pairs of adjoint modalities (♦, □) and (♦, □), with no explicit relationship between the modalities of the same colour, namely, (♦, □) and (♦, □). The modalities form an adjunction as follows: $A \rightarrow □B$ if ♦$A \rightarrow B$ and $A \rightarrow □B$ if ♦$A \rightarrow B$.

Our shallow inference calculus LBiKt is a merger of two sub-calculi for BiInt and Kt derived from Belnap’s inherently modular display logic. LBiKt has syntactic cut-elimination, but is ill-suited for backward proof search. Our deep inference calculus DBiKt is complete with respect to the cut-free fragment of LBiKt and is more amenable to proof search as it contains no display postulates and contraction rules. To complete the picture, we also give a Kripke semantics for BiKt based upon three relations ≤, $R_♦$ and $R_□$. The logic BiKt enjoys various desirable properties:

* **Conservativity:** it is a conservative extension of intuitionistic logic Int, dual intuitionistic logic DInt, and bi-intuitionistic logic BiInt;
* **Classical Collapse:** it collapses to classical tense logic by the addition of four structural rules;
* **Disjunction Property:** If $A \lor B$ is a theorem not containing $\rightarrow$ then $A$ is a theorem or $B$ is a theorem;
* **Dual Disjunction Property:** If $A \land B$ is a counter-theorem not containing $\rightarrow$ then so is $A$ or $B$;
* **Independent ♦ and □:** there is no a priori relationship between these connectives.

The independence of ♦ and □ is a departure from traditional intuitionistic tense or modal logics, e.g., those considered by Ewald [6] and Simpson [24]. Both Ewald and Simpson allow a form of interdependency between ♦ and □, expressed as the axiom $(♦A \rightarrow □B) \rightarrow □(A \rightarrow B)$, which is not derivable in LBiKt. However, we shall see in Section 7 that we can recover Ewald’s intuitionistic tense logic and Simpson’s intuitionistic modal logic by extending LBiKt with two structural rules.

Due to space limit, some proofs are omitted, but they can be found in an extended version of this paper [11].
\[ \begin{align*}
\tau^-(A) &= A & \tau^+(A) &= A \\
\tau^-(X, Y) &= \tau^-(X) \land \tau^-(Y) & \tau^+(X, Y) &= \tau^+(X) \lor \tau^+(Y) \\
\tau^-(X \triangleright Y) &= \tau^-(X) \prec \tau^+(Y) & \tau^+(X \triangleright Y) &= \tau^-(X) \rightarrow \tau^+(Y) \\
\tau^-(\bigcirc X) &= \circ \tau^-(X) & \tau^+(\bigcirc X) &= \Box \tau^+(X) \\
\tau^-(\bullet X) &= \bullet \tau^-(X) & \tau^+(\bullet X) &= \blacksquare \tau^+(X) \\
\tau(X \triangleright Y) &= \tau^-(X) \rightarrow \tau^+(Y)
\end{align*} \]

Fig. 1. Formula Translation of Nested Sequents

2 Nested Sequents

The formulae of BiKt are built from a set \( \text{Atoms} \) of atomic formulae via the grammar below, with \( p \in \text{Atoms} \):

\[ A ::= p \mid \top \mid \bot \mid A \rightarrow A \mid A \leftarrow A \mid A \land A \mid A \lor A \mid \Box A \mid \Diamond A \mid \blacksquare A. \]

A structure is defined by the following grammar, where \( A \) is a BiKt formula:

\[ X ::= \emptyset \mid A \mid (X, X) \mid X \triangleright X \mid \bigcirc X \mid \bullet X. \]

The structural connective \( \cdot \), is associative and commutative and \( \emptyset \) is its unit.

We always consider structures modulo these equivalences. To reduce parentheses, we assume that \( \bigcirc \) and \( \bullet \) bind tighter than \( \cdot \), which binds tighter than \( \triangleright \).

Thus, we write \( \bullet X, Y \triangleright Z \) to mean \((\bullet(X), Y) \triangleright Z\).

A nested sequent is a structure of the form \( X \triangleright Y \). This notion of nested sequents generalises Kashima’s nested sequents [13] for classical tense logics, Brunnler’s nested sequents [2] and Poggiolesi’s tree-hypersequents [19] for classical modal logics. Figure 1 shows the formula-translation of nested sequents. On both sides of the sequent, \( \bigcirc \) is interpreted as a white (modal) operator and \( \bullet \) as a black (tense) operator. Note that however, on the lefthand side of the sequent, \( \cdot \) is interpreted as exclusion, while on the righthand side, it is interpreted as implication.

A context is a structure with a hole or a placeholder \([\cdot]\). Contexts are ranged over by \( \Sigma[\cdot] \). We write \( \Sigma[\cdot][X] \) for the structure obtained by filling the hole \([\cdot]\) in the context \( \Sigma[\cdot] \) with a structure \( X \).

A simple context is defined via:

\[ \Sigma[\cdot] ::= \emptyset \mid \Sigma[\cdot], Y \mid (Y, \Sigma[\cdot]) \mid \circ \Sigma[\cdot] \mid \bullet \Sigma[\cdot] \]

Intuitively, the hole in a simple context is never under the scope of \( \triangleright \). Positive and negative contexts are defined inductively as follows:

* If \( \Sigma[\cdot] \) is a simple context then \( \Sigma[\cdot][\emptyset] \triangleright Y \) is a negative context and \( Y \triangleright \Sigma[\cdot] \) is a positive context.

* If \( \Sigma[\cdot] \) is a positive/negative context then so are \((\Sigma[\cdot], Y), (Y, \Sigma[\cdot]), \circ(\Sigma[\cdot]), \bullet(\Sigma[\cdot]), \Sigma[\cdot] \triangleright Y, \) and \( Y \triangleright \Sigma[\cdot] \).

We write \( \Sigma^- [\cdot] \) to indicate that \( \Sigma[\cdot] \) is a negative context and \( \Sigma^+ [\cdot] \) to indicate that it is a positive context. Intuitively, if one views a nested sequent as a tree (with structural connectives and formulae as nodes), then a hole in a context is negative (positive) if it appears to the left (right) of the closest ancestor node labelled with \( \triangleright \).
As a consequence of the overloading of $\triangleright$ as a structural proxy for both $\rightarrow$ and $\prec$, further nesting of a positive/negative context within $\triangleright$ does not change its polarity. This is different from the traditional notion of polarity which is defined in terms of either $\rightarrow$ or $\prec$ alone, but not both. This aspect is different from display calculi and may cause confusion at first reading. Our statement of the display property in Lemma 3.2–Lemma 3.2 accounts for this difference.

The context $\Sigma\{\}$ is strict if it has any of the forms:

\[
\Sigma'[X \triangleright \{ \}] \quad \Sigma'[\{ \} \triangleright X] \quad \Sigma'[\circ \{ \}] \quad \Sigma'[^{•} \{ \}]
\]

Intuitively, in the formation tree of a strict context, the hole must be an immediate child of $\triangleright$ or $\circ$ or $^{•}$. This notion of strict contexts will be used in later in Section 3.

Example 2.1 The context $^{•}([\{., (X \triangleright Y) \triangleright Z])$ is a simple context but $^{•}([\{., (X \triangleright Y) \triangleright Z]$ is not. Both $^{•}([\{., (X \triangleright Y) \triangleright Z$ and $^{•}((\{., X \triangleright Y) \triangleright Z$ are negative contexts. The context $^{•}([\{., X \triangleright Y) \triangleright Z$ is not.

3 Nested Sequent Calculi

We now present the two nested sequent calculi that we will use in the rest of the paper: a shallow inference calculus $\text{LBiKt}$ and a deep inference calculus $\text{DBiKt}$. Fig. 2 gives the rules of the shallow inference calculus $\text{LBiKt}$. The inference rules of $\text{LBiKt}$ can only be applied to formulae at the top level of nested sequents, and the structural rules $s_L$, $s_R$, $\triangleright_L$, $\triangleright_R$, $rp_\circ$ and $rp_{•}$, also called the residuation rules, are used to bring the required sub-structures to the top level. These rules are similar to residuation postulates in display logic, are essential for the cut-elimination proof of $\text{LBiKt}$, but contain too much non-determinism for effective proof search. Another issue with proof search in $\text{LBiKt}$ is the structural contraction rules, which allow contraction on arbitrary structures, not just formulae as in traditional sequent calculi. $\text{LBiKt}$ is as a merger of two calculi: the $\text{LBIInt}$ calculus [9,20] for the intuitionistic connectives, and the display calculus [8] for the tense connectives.

We use $\circ$ and $^{•}$ as structural proxies for the non-residuated pairs ($\Diamond$, $\Box$) and ($\Diamond$, $\blacksquare$) respectively, whereas Wansing [25] uses only $^{•}$ as a structural proxy for the residuated pair ($\Diamond$, $\Box$) and recovers ($\Diamond$, $\blacksquare$) via classical negation, while Goré [8] uses $\circ$ and $^{•}$ as structural proxies for the residuated pairs ($\Diamond$, $\blacksquare$) and ($\Diamond$, $\Box$) 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.

Fig. 3 gives the rules of the deep inference calculus $\text{DBiKt}$. 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 $\text{LBiKt}$ can be simulated by deep inference and propagation rules in $\text{DBiKt}$. Another feature of $\text{DBiKt}$ 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 $\Sigma$ which itself contains a formula $A$ and a $^{•}$-structure, such that the $^{•}$-structure contains $\Box A$.

$\text{DBiKt}$ achieves the goal of merging the $\text{LBIInt}$ calculus [20] and a two-sided
Identity and logical constants:
\[
\begin{align*}
&X, A \vdash A, Y & \text{id} \\
&X, \bot \vdash Y & \bot_L \\
&X \vdash \top, Y & \top_R
\end{align*}
\]

Structural rules:
\[
\begin{align*}
&\frac{X \vdash Z}{X, Y \vdash Z} w_L \\
&\frac{X \vdash Z}{X \vdash Y, Z} w_R \\
&\frac{(X_1 \vdash Y_1), X_2 \vdash Y_2}{X_1, X_2 \vdash Y_1, Y_2} s_L \\
&\frac{X_1 \vdash Y_1, (X_2 \vdash Y_2)}{X_1, X_2 \vdash Y_1, Y_2} s_R \\
&\frac{X \vdash Y}{\bullet \vdash Y} r_{p_{\circ}} \\
&\frac{\circ \vdash Y}{X \vdash \bullet \vdash Y} r_{p_{\bullet}} \\
&\frac{X_1 \vdash Y_1, A, X_2 \vdash Y_2}{X_1, X_2 \vdash Y_1, Y_2} \text{cut}
\end{align*}
\]

Logical rules:
\[
\begin{align*}
&\frac{X, B_1 \vdash Y}{X, B_1 \land B_2 \vdash Y} \land_L i \in \{1, 2\} \\
&\frac{X, A \vdash Y, X, B \vdash Y}{X, A \land B \vdash Y} \land_R \\
&\frac{X \vdash B_1, Y}{X \vdash B_1 \lor B_2, Y} \lor_L i \in \{1, 2\} \\
&\frac{X \vdash A, Y}{X \vdash A \land B, Y} \lor_R \\
&\frac{X \vdash A \rightarrow B, Y}{X \vdash B \rightarrow A, Y} \rightarrow_L \\
&\frac{X \vdash A}{X, A \rightarrow B \vdash Y} \rightarrow_R \\
&\frac{X \vdash A}{X \vdash A \land B, Y} \neg_L \\
&\frac{X \vdash A, X}{X \vdash A \land \neg B, Y} \neg_R
\end{align*}
\]

\[
\begin{align*}
&\frac{\square \vdash X}{\square \vdash \circ X} \square_L \\
&\frac{X \vdash \circ A, X}{\circ X \vdash \square A} \square_R \\
&\frac{\circ A \vdash X}{\circ A \vdash \circ A} \circ_L \\
&\frac{A \vdash \circ A}{\circ A \vdash \circ A} \circ_R \\
&\frac{A \vdash X}{\circ A \vdash X} \circ_R \\
&\frac{\bullet \vdash A, X}{\circ \vdash A, \circ \vdash X} \bullet_L \\
&\frac{\bullet \vdash A, X}{\circ \vdash A, \circ \vdash X} \bullet_R \\
&\frac{A \vdash X}{\circ A \vdash X} \circ_R
\end{align*}
\]

Fig. 2. LBiKt: a shallow inference system for BiKt

version of the DKt calculus [10]. While in the shallow inference case, a calculus for BiKt could be obtained relatively easily by merging shallow inference calculi for BiInt and tense logics, the combination of calculi is not so obvious in the deep inference case. Although the propagation rules for \(\triangleright\)-structures remain the same as in the BiInt case [20], the propagation rules for \(\circ\)- and \(\bullet\)-structures are not as simple as in the DKt calculus [10]. Since we do not assume any direct relationship between \(\square\) and \(\Diamond\), or \(\blacksquare\) and \(\spadesuit\), propagation rules like \(\blacksquare_{L2}\) need to involve the \(\triangleright\) structural connective so they can refer to both sides of the nested sequent.

Note that in the rules \(\rightarrow_L\) and \(\neg_R\) in DBiKt, we require that the contexts in which the principal formulae reside are strict contexts. This is strictly speaking not necessary, i.e., we could remove the proviso without affecting the expressivity of the proof system. The proviso does, however, reduce the non-determinism in partitioning the contexts in \(\rightarrow_L\) or \(\neg_R\). Consider, for example, the nested sequent \(\circ(a, b \rightarrow c) \triangleright d\). Without the requirement of strict contexts, there are two instances of \(\rightarrow_L\) with that nested sequent as the conclusion:

\[
\frac{\circ(a, (b \rightarrow c) \triangleright b), \circ(\circ(a, (b \rightarrow c)) \triangleright d) \triangleright d}{\circ(a, b \rightarrow c) \triangleright d} \rightarrow_L
\]

5
Identity and logical constants:

\[
\begin{align*}
\Sigma[X, A \vdash A, Y] & \quad \vdash_L \\
\Sigma[\bot, X \vdash Y] & \quad \vdash_R \\
\Sigma[X \vdash \top, Y] & \quad \vdash_R
\end{align*}
\]

Propagation rules:

\[
\begin{align*}
\Sigma^-[A, (A, X \vdash Y)] & \quad \vdash_{L1} \\
\Sigma^+[X \vdash Y, A] & \quad \vdash_{R1} \\
\Sigma[X, A \triangleright W, (A, Y \triangleright Z)] & \quad \vdash_{L2} \\
\Sigma[(X \triangleright Y), W \triangleright A, Z] & \quad \vdash_{R2}
\end{align*}
\]

\[
\begin{align*}
\Sigma^-[A \circ (\bullet A, X)] & \quad \vdash_{L1} \\
\Sigma^+[A, \circ (\bullet A, X)] & \quad \vdash_{R1}
\end{align*}
\]

\[
\begin{align*}
\Sigma^+[A, \bowtie (\bullet A, X)] & \quad \vdash_{R2}
\end{align*}
\]

Logical rules:

\[
\begin{align*}
\Sigma^-[A \lor B, A] & \quad \vdash_{L1} \\
\Sigma^+[A \lor B] & \quad \vdash_{R1} \\
\Sigma^-[A \land B, A, B] & \quad \vdash_{L2} \\
\Sigma^+[A \land B] & \quad \vdash_{R2} \\
\Sigma^-[A \leftarrow B, (A \bowtie B)] & \quad \vdash_{L3} \\
\Sigma^+[A \leftarrow B] & \quad \vdash_{R3} \\
\Sigma^+[X, A \rightarrow B, A] & \quad \vdash_{L4} \\
\Sigma^-[X, A \rightarrow B] & \quad \vdash_{R4} \\
\Sigma^+[X, A \leftarrow B, A] & \quad \vdash_{L5} \\
\Sigma^-[X, A \leftarrow B] & \quad \vdash_{R5}
\end{align*}
\]

\[
\begin{align*}
\Sigma^-[A \circ (\circ A, A)] & \quad \vdash_{L6} \\
\Sigma^+[A \circ (\circ A, A)] & \quad \vdash_{R6}
\end{align*}
\]

Fig. 3. DBiKt: a deep inference system for BiKt

\[
\begin{align*}
\circ(a, b \rightarrow c, d) & \quad d \quad \circ(a, b \rightarrow c, d) & \quad d \\
\circ(a, b \rightarrow c) & \quad d \\
\end{align*}
\]

In the first instance, the context is \(\circ(a, [\ ] \triangleright d)\), which is not strict, whereas in the second instance, it is \(\circ([\ ] \triangleright d)\), which is strict. In general, if there are \(n\) formulae connected to \(b \rightarrow c\) via the comma structural connective, then there are \(2^n\) possible instances of \(\rightarrow\), without the strict context proviso.

We write \(\vdash_{LBiKt} \pi : X \triangleright Y\) when \(\pi\) is a derivation of the shallow sequent \(X \triangleright Y\) in \(LBiKt\), and \(\vdash_{DBiKt} \pi : X \triangleright Y\) when \(\pi\) is a derivation of the sequent \(X \triangleright Y\) in \(DBiKt\). In either calculus, the height \(|\pi|\) of a derivation \(\pi\) is the number of sequents on the longest branch.

**Example 3.1** Below we derive Ewald’s axiom 9 for \(IK_1[6]\) in \(LBiKt\) and \(DBiKt\). The \(LBiKt\)-derivation on the left read bottom-up brings the required sub-structure
Goré, Postniece and Tiu

♦ A to the top-level using the residuation rule \( rp_0 \) and applies ♦ _R backward. The DBiKt-derivation on the right instead applies □ _R deeply, and propagates the required formulae to the appropriate sub-structure using ♦ _R1. Note that contraction is implicit in ♦ _R1, and all propagation rules.

\[
\begin{align*}
\frac{\emptyset \vdash A & \quad \text{id}}{\bullet \vdash A \quad \text{rp}_0} \\
\frac{\vdash \bullet \vdash A \quad \text{rp}_0}{A \vdash \bullet \vdash A \quad \text{rp}_0} \\ 
\frac{\vdash \bullet \vdash A \quad \text{rp}_0}{\emptyset \vdash A \rightarrow \bullet \vdash A \quad \text{rp}_0}
\end{align*}
\]

Display property

A (deep or shallow) nested sequent can be seen as a tree of traditional sequents. The structural rules of LBiKt allows 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. We shall use two “display” rules which are easily derivable using \( s_L, s_R, \triangleright L \) and \( \triangleright R \):

\[
\begin{align*}
\frac{(X_1 \triangleright X_2) \triangleright Y}{X_1 \triangleright X_2, Y \quad \text{rp}_L^L} \\
\frac{X_1 \triangleright (X_2 \triangleright Y)}{X_1, X_2 \triangleright Y \quad \text{rp}_L^R}
\end{align*}
\]

Let \( DP = \{ \text{rp}_L^L, \text{rp}_L^R, \text{rp}_0, \text{rp}_\bullet \} \) and let \( DP \)-derivable mean “derivable using rules only from \( DP \)”.

**Lemma 3.2 (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:

(i) \( Y \triangleright p \) is \( DP \)-derivable from \( X \triangleright \Sigma[p] \) and

(ii) \( p \triangleright Z \) is \( DP \)-derivable from \( \Sigma[p] \triangleright X \).

**Lemma 3.3 (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:

(i) \( Y \triangleright p \) is \( DP \)-derivable from \( X \triangleright \Sigma[p] \), and

(ii) \( Z \triangleright p \) is \( DP \)-derivable from \( \Sigma[p] \triangleright X \).

**Lemma 3.4 (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:

(i) \( p \triangleright Y \) is \( DP \)-derivable from \( X \triangleright \Sigma[p] \) and

(ii) \( p \triangleright Z \) is \( DP \)-derivable from \( \Sigma[p] \triangleright X \).

Since the rules in \( DP \) are all invertible, the derivations constructed in the above lemmas are invertible derivations. That is, we can derive \( Y \triangleright p \) from \( X \triangleright \Sigma[p] \) and vice versa. Note also that since rules in the shallow system are closed under substitution, this also means \( Y \triangleright Z \) is derivable from \( X \triangleright \Sigma[Z] \), and vice versa, for any \( Z \).
Cut elimination in LBiKt

Our cut-elimination proof is based on the method of proof-substitution presented in [9]. It is very similar to the general cut elimination method used in display calculi. The proof relies on the display property and the fact that inference rules in LBiKt are closed under substitutions. The proof is omitted here, but is available in the extended version of this paper [11]. We illustrate one case with an example.

Consider the derivation below ending with a cut on ♦A:

\[ \Pi_1, X \triangleright Y, \Diamond A \]

Instead of permuting the cut rule locally, we trace the cut formula ♦A until it becomes principal in the derivations \(\Pi_1\) and \(\Pi_2\), and then apply cut on a smaller formula. Suppose that \(\Pi_1\) and \(\Pi_2\) are respectively the derivations (1) and (2) in Figure 4. We first transform \(\Pi_1\) by substituting \((X_2 \triangleright Y_2)\) for ♦A in \(\Pi_1\) and obtain the sub-derivation with an open leaf as shown in Figure 4(3). We then prove the open leaf by uniformly substituting \(\Diamond(X'_1)\) for ♦A in \(\Pi_2\), and applying cut on a sub-formula A, as shown in Figure 4(4).

**Theorem 3.5** If \(X \triangleright Y\) is LBiKt-derivable then it is also LBiKt-derivable without using cut.

4 Equivalence between LBiKt and DBiKt

We now show that LBiKt and DBiKt are equivalent. We first show that every derivation in DBiKt can be mimicked by a cut-free derivation in LBiKt. The interesting 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 4.1** For any \(X\) and \(Y\), if \(\vdash_{DBiKt} \pi : X \triangleright Y\) then \(\vdash_{LBiKt} \pi' : X \triangleright Y\).

**Proof.** (Outline) We show that each deep inference rule \(\rho\) is derivable in the shallow
system by a case analysis of the context $\Sigma[\ ]$ in which the deep rule $\rho$ applies. Note that if a deep inference rule $\rho$ 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 valid instance of $\rho$ where $\Sigma[\ ] = [\ ]$ is derivable in the shallow system. For the case where $\Sigma[\ ]$ is either positive or negative, we use the display property. We show here the case where $\rho$ is a rule with a single premise; the other cases are similar. Suppose $\rho$ is as shown below left. By the display properties, we need to show only that the rule on the right is derivable in the shallow system for some structure $X'$:

$$\begin{align*}
\frac{\Sigma^+[U]}{\Sigma^+[V]} & \quad \rho \\
\frac{X' \triangleright U}{X' \triangleright V}
\end{align*}$$

A more detailed proof is given in the extended paper. $\dashv$

We now show that any cut-free $\text{LBiKt}$-derivation can be transformed into a cut-free $\text{DBiKt}$-derivation. This requires proving cut-free admissibility of various structural rules in $\text{DBiKt}$. The admissibility of general weakening and formula contraction (not general contraction, which we will show later) are straightforward by induction on the height of derivations.

**Lemma 4.2 (Admissibility of general weakening)** For any structures $X$ and $Y$: if $\vdash_{\text{DBiKt}} \pi : \Sigma[X]$ then $\vdash_{\text{DBiKt}} \pi' : \Sigma[X, Y]$ such that $|\pi'| \leq |\pi|$.

**Lemma 4.3 (Admissibility of formula contraction)** For any structure $X$ and formula $A$: if $\vdash_{\text{DBiKt}} \pi : \Sigma[X, A, A]$ then $\vdash_{\text{DBiKt}} \pi' : \Sigma[X, A]$ such that $|\pi'| \leq |\pi|$.

Once weakening and contraction are shown admissible, it remains to show that the residuation rules of $\text{LBiKt}$ are also admissible. In contrast to the case with the deep inference system for bi-intuitionistic logic, the combination of modal and intuitionistic structural connectives complicates the proof of this admissibility. It seems crucial to first show “deep” admissibility of certain forms of residuation for $\triangleright$. We state the required lemmas below.

**Lemma 4.4 (Deep admissibility of structural rules)** The following statements hold for $\text{DBiKt}$:

(i) Deep admissibility of $s_L$. If $\vdash_{\text{DBiKt}} \pi : \Sigma[(X \triangleright Y), Z \triangleright W]$ then $\vdash_{\text{DBiKt}} \pi' : \Sigma[X, Z \triangleright Y, W]$ such that $|\pi'| \leq |\pi|$.

(ii) Deep admissibility of $s_R$. If $\vdash_{\text{DBiKt}} \pi : \Sigma[X \triangleright Y, (Z \triangleright W)]$ then $\vdash_{\text{DBiKt}} \pi' : \Sigma[X, Z \triangleright Y, W]$ such that $|\pi'| \leq |\pi|$.

(iii) Deep admissibility of $\triangleright_L$. If $\vdash_{\text{DBiKt}} \pi : \Sigma[X \triangleright Y, Z]$ and $\Sigma$ is either the empty context $[\ ]$ or a negative context $\Sigma^+_1[\ ]$, then $\vdash_{\text{DBiKt}} \pi' : \Sigma[(X \triangleright Y) \triangleright Z]$.

(iv) Deep admissibility of $\triangleright_R$. If $\vdash_{\text{DBiKt}} \pi : \Sigma[X, Y \triangleright Z]$ and $\Sigma$ is either the empty context $[\ ]$ or a positive context $\Sigma^+_1[\ ]$, then $\vdash_{\text{DBiKt}} \pi' : \Sigma[X \triangleright (Y \triangleright Z)]$.

We now show that the residuation rules of $\text{LBiKt}$ for $\circ$- and $\bullet$-structures are admissible in $\text{DBiKt}$, i.e., they can be simulated by the propagation rules of $\text{DBiKt}$.

**Lemma 4.5 (Admissibility of residuation)** The following statements hold in $\text{DBiKt}$:
Admissibility of $rp\circ$. If $\vdash_{DBiKt} \pi : X \triangleright Z$ then $\vdash_{DBiKt} \pi' : X \triangleright \circ Z$.

Admissibility of $rp\circ$. If $\vdash_{DBiKt} \pi : \circ X \triangleright Z$ then $\vdash_{DBiKt} \pi' : X \triangleright \circ Z$.

Admissibility of $rp\circ$. If $\vdash_{DBiKt} \pi : X \triangleright \circ Z$ then $\vdash_{DBiKt} \pi' : X \triangleright \circ Z$.

Admissibility of $rp\circ$. If $\vdash_{DBiKt} \pi : \circ X \triangleright Z$ then $\vdash_{DBiKt} \pi' : X \triangleright \circ Z$.

The proof of admissibility of general contraction is more involved and requires proving several distribution properties among structural connectives. The proof can be found in Appendix A.

Lemma 4.6 (Admissibility of general contraction) For any structure $Y$: if $\vdash_{DBiKt} \pi : \Sigma[Y,Y]$ then $\vdash_{DBiKt} \pi' : \Sigma[Y]$.

Once all structural rules of $LBiKt$ are shown admissible in $DBiKt$, it is easy to show that every derivation in $LBiKt$ can be translated to a derivation in $DBiKt$.

Theorem 4.7 For any $X$ and $Y$, if $\vdash_{LBiKt} \pi : X \triangleright Y$ then $\vdash_{DBiKt} \pi' : X \triangleright Y$.

Corollary 4.8 For any $X$ and $Y$, $\vdash_{LBiKt} \pi : X \triangleright Y$ if and only if $\vdash_{DBiKt} \pi' : X \triangleright Y$.

Proof. By Theorems 4.1 and 4.7. \(\square\)

5 Proof Search

In this section we outline a proof search strategy for $DBiKt$, closely following the approaches presented in [20] and [10]. 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 backward 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 backward 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., $\rightarrow_{R}$).

A context $\Sigma[ ]$ is said to be headed by a structural connective $\#$ if the topmost symbol in the formation tree of $\Sigma[ ]$ is $\#$. A context $\Sigma[ ]$ is said to be a factor of $\Sigma'[ ]$ if $\Sigma[ ]$ is a subcontext of $\Sigma'[ ]$ and $\Sigma[ ]$ is headed by either $\triangleright$, $\circ$ or $\bullet$. We write $\hat{\Sigma}[ ]$ to denote the smallest factor of $\Sigma[ ]$. We write $\hat{\Sigma}[X]$ to denote the structure $\Sigma_1[X]$, if $\Sigma_1[ ] = \hat{\Sigma}[ ]$. We define the top-level formulae of a structure as:

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

For example, if $\Sigma[ ] = (A, B \triangleright C, \bullet (D, (E \triangleright F) \triangleright [ ]))$, then $\hat{\Sigma}[G] = (D, (E \triangleright F) \triangleright G)$, and $\{D, (E \triangleright F)\} = \{D\}$.

Let $\triangleleft L_1$ and $\rightarrow R_1$ denote two new derived rules (see [20] for their derivation):

$$\frac{\Sigma\gamma [A, A \triangleleft B]}{\Sigma\gamma [A \triangleleft B]} \triangleleft L_1 \quad \frac{\Sigma\gamma [A \rightarrow B, B]}{\Sigma\gamma [A \rightarrow B]} \rightarrow R_1$$
We now define a notion of a saturated structure, which is similar to that of a traditional sequent. Note that we need to define it for both structures headed by ▷ and those headed by ◦ or •. A structure \( X ▷ Y \) is saturated if it satisfies the following:

1. \( \{X\} \cap \{Y\} = \emptyset \)
2. If \( A \land B \in \{X\} \) then \( A \in \{X\} \) and \( B \in \{Y\} \)
3. If \( A \land B \in \{Y\} \) then \( A \in \{Y\} \) or \( B \in \{X\} \)
4. If \( A \lor B \in \{X\} \) then \( A \in \{X\} \) or \( B \in \{X\} \)
5. If \( A \lor B \in \{Y\} \) 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 \( A \leftarrow B \in \{Y\} \) then \( A \in \{Y\} \) or \( B \in \{X\} \)
8. If \( A \leftarrow B \in \{X\} \) then \( A \in \{X\} \)
9. If \( A \rightarrow B \in \{Y\} \) then \( B \in \{Y\} \)

Dually, \( \diamond Y \) or • is right-saturated if it satisfies (3), (5), (9) above, and

6' If \( A \rightarrow B \in \{X\} \) then \( B \in \{X\} \).

We now define a notion of a saturated structure, which is similar to that of a traditional sequent. Note that we need to define it for both structures headed by ▷ and those headed by ◦ or •. A structure \( X ▷ Y \) is saturated if it satisfies the following:

1. \( \{X\} \cap \{Y\} = \emptyset \)
2. If \( A \land B \in \{X\} \) then \( A \in \{X\} \) and \( B \in \{Y\} \)
3. If \( A \land B \in \{Y\} \) then \( A \in \{Y\} \) or \( B \in \{X\} \)
4. If \( A \lor B \in \{X\} \) then \( A \in \{X\} \) or \( B \in \{X\} \)
5. If \( A \lor B \in \{Y\} \) 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 \( A \leftarrow B \in \{Y\} \) then \( A \in \{Y\} \) or \( B \in \{X\} \)
8. If \( A \leftarrow B \in \{X\} \) then \( A \in \{X\} \)
9. If \( A \rightarrow B \in \{Y\} \) then \( B \in \{Y\} \)

For structures of the form ◦X or •X, we need to define two notions of saturation, left saturation and right saturation. The former is used when ◦X is nested in a negative context, and the latter when it is in a positive context. A structure ◦X or •X is left-saturated if it satisfies (2), (4), (8) above, and

6' If \( A \rightarrow B \in \{X\} \) then \( B \in \{X\} \).

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, \( (A ▷ B) \in (A, (A ▷ B), ◦C) \). The realisation of formulae by a structure \( X \) is defined as follows:

* \( A ▷ B \) (\( A \leftarrow B \), resp.) is right-realised (resp. left-realised) by \( X \) iff there exists \( Z ▷ W \in X \) such that \( A \in \{Z\} \) and \( B \in \{W\} \).
* \( \square A \) (\( ◦ A \), resp.) is right-realised (resp. left-realised) by \( X \) iff there exists \( ◦(Z ▷ W) \in X \) or \( ◦W \in X \) (resp. \( ◦(W ▷ Z) \in X \) or \( ◦W \in X \)) such that \( A \in \{W\} \).
* \( ▷ A \) (\( • A \), resp.) is right-realised (resp. left-realised) by \( X \) iff there exists \( •(W ▷ Z) \in X \) or \( •Z \in X \) (resp. \( •(Z ▷ W) \in X \) or \( •Z \in X \)) such that \( A \in \{Z\} \).

We say that a structure \( X \) is left-realised iff every formula in \( \{X\} \) with top-level connective \( \leftarrow, ◦ \) or • is left-realised by \( X \). Right-realisation of \( X \) is defined dually. We say that a structure occurrence \( X \) in \( \Sigma[A] \) 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:

* \( X_1 ▷ Y_1 \supset X_0 ▷ Y_0 \) iff \( \{X_1\} \supset \{X_0\} \) or \( \{Y_1\} \supset \{Y_0\} \).
* \( ◦X \supset ◦Y \) iff \( •X \supset •Y \) iff \( \{X\} \supset \{Y\} \).

To simplify presentation, we use the following terminology: Given a structure \( \Sigma[A] \), we say that \( \Sigma[A] \) is saturated if \( \Sigma[A] \) is \( X ▷ Y \) and it is saturated; or \( \Sigma[A] \) is either \( ◦X \) or •X and it is either left- or right-saturated (depending on its position in \( \Sigma[A] \)). We say that \( \Sigma[A] \) is propagated if its occurrence in \( \Sigma[A] \) is propagated,
and we say that $A$ is realised by $\hat{\Sigma}[A]$, if either
* $\hat{\Sigma}[A] = (X \triangleright Y)$ and either $A \in \{X\}$ is left-realised by $X$, or $A \in \{Y\}$ is right realised by $Y$; or
* $\hat{\Sigma}[A]$ is either $\circ X$ or $\bullet X$, and, depending on the polarity of $\Sigma[ ]$, $A$ is either left- or right-realised by $X$.

We now outline an approach to proof search in DBiKt. We approach this by modifying DBiKt to obtain a calculus DBiKt\textsubscript{1} that is more amenable to proof search. Our approach follows that of our previous work on bi-intuitionistic logic \cite{20} 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 $\rightarrow_R$ rule (see condition ii of Definition 5.1). Additionally and more importantly, our proof search calculus addresses the issue that some modal propagation rules of DBiKt, e.g. $\Box_{L2}$, create $\triangleright$-structures during backward proof search. This property of DBiKt is undesirable and gives rise to non-termination if rules like $\Box_{L2}$ are applied naively.

**Definition 5.1** Let DBiKt\textsubscript{1} be the system obtained from DBiKt with the following changes:

(i) Add the derived rules $\prec L1$ and $\rightarrow R1$.

(ii) Restrict rules $\prec L1$ with $\rightarrow R$ with the following condition: the rule is applicable only if $\hat{\Sigma}[A\#B]$ is saturated and propagated, and $A\#B$ is not realised by $\hat{\Sigma}[A\#B]$, for $\# \in \{\rightarrow, \prec \}$.

(iii) Replace rules $\triangleright L1$ and $\triangleright R1$ with the following:

\[
\frac{\Sigma[A,(A, X \triangleright Y), W \triangleright Z]}{\Sigma[(A, X \triangleright Y), W \triangleright Z]} \triangleright L1 \quad \frac{\Sigma[W \triangleright Z,(X \triangleright Y, A)]}{\Sigma[W \triangleright Z,(X \triangleright Y, A)]} \triangleright R1
\]

(iv) Restrict rules $\triangleright L2$ and $\triangleright R2$ with the following condition: the rule is applicable only if $A \notin \{Y\}$.

(v) Replace rules $\Diamond L$, $\Box R$, $\lozenge L$, $\blacksquare R$ with the following, where the rule is applicable only if $\Sigma[\#A]$ is saturated and propagated and $\#A$ is not realised by $\hat{\Sigma}[\#A]$, for $\# \in \{\Diamond, \Box, \lozenge, \blacksquare\}$:

\[
\frac{\Sigma^-[\Diamond A, \circ(A \triangleright \emptyset)]}{\Sigma^-[\Diamond A]} \Diamond L \quad \frac{\Sigma^+[\Box A, \circ(\emptyset \triangleright A)]}{\Sigma^+[\Box A]} \Box R
\]

\[
\frac{\Sigma^-[\lozenge A, \bullet(A \triangleright \emptyset)]}{\Sigma^-[\lozenge A]} \lozenge L \quad \frac{\Sigma^+[\blacksquare A, \bullet(\emptyset \triangleright A)]}{\Sigma^+[\blacksquare A]} \blacksquare R
\]

(vi) Replace rules $\blacksquare L2$, $\Box_{L2}$ with the following, where $A \notin \{Y_1\}$:

\[
\frac{\Sigma[\blacksquare A, X \triangleright \bullet(A, Y_1 \triangleright Y_2), Z]}{\Sigma[\blacksquare A, X \triangleright \bullet(Y_1 \triangleright Y_2), Z]} \blacksquare L2 \quad \frac{\Sigma[\Box A, X \triangleright \circ(A, Y_1 \triangleright Y_2), Z]}{\Sigma[\Box A, X \triangleright \circ(Y_1 \triangleright Y_2), Z]} \Box_{L2}
\]

(vii) Replace rules $\Diamond_{R2}$, $\lozenge_{R2}$ with the following, where $A \notin \{X_2\}$:

\[
\frac{\Sigma[\Diamond(X_1 \triangleright X_2, A), Y \triangleright Z, \Diamond A]}{\Sigma[\Diamond(X_1 \triangleright X_2, A), Y \triangleright Z]} \Diamond_{R2} \quad \frac{\Sigma[\lozenge(X_1 \triangleright X_2, A), Y \triangleright Z, \lozenge A]}{\Sigma[\lozenge(X_1 \triangleright X_2, A), Y \triangleright Z]} \lozenge_{R2}
\]
(viii) Replace rules □_L, ◁_R, □_L, ◁_R with the following:

\[
\begin{align*}
\Sigma^- [A, \bullet (\square A, X \triangleright Y)] & \quad \Rightarrow \quad \square L_1 \\
\Sigma^- [\bullet (\square A, X \triangleright Y)] & \quad \Rightarrow \quad \square L_1 \\
\Sigma^+ [A, \bullet (Y \triangleright A, X)] & \quad \Rightarrow \quad \triangle R_1 \\
\Sigma^+ [\bullet (Y \triangleright A, X)] & \quad \Rightarrow \quad \triangle R_1 \\
\Sigma^- [A, \circ (\square A, X \triangleright Y)] & \quad \Rightarrow \quad \square L_1 \\
\Sigma^- [\circ (\square A, X \triangleright Y)] & \quad \Rightarrow \quad \square L_1 \\
\Sigma^+ [A, \circ (Y \triangleright A, X)] & \quad \Rightarrow \quad \triangle R_1 \\
\Sigma^+ [\circ (Y \triangleright A, X)] & \quad \Rightarrow \quad \triangle R_1
\end{align*}
\]

(ix) Replace rules →_L, \prec_R with the following:

\[
\begin{align*}
\Sigma [X, A \rightarrow B \triangleright A, Y] & \quad \Rightarrow \quad \Sigma [X, A \rightarrow B \triangleright Y] \rightarrow L \\
\Sigma [X \triangleright Y, A \prec B, A] & \quad \Rightarrow \quad \Sigma [X, B \triangleright Y, A \prec B] \prec R \\
\Sigma [X \triangleright Y, A \prec B] & \quad \Rightarrow \quad \Sigma [X \triangleright Y, A \prec B]
\end{align*}
\]

(x) Restrict rules →_L, \prec_R, →_L, \preceq_R, \wedge_L, \wedge_R, \vee_L, \vee_R and all modal propagation rules to the following: Let Σ[X_0] be the conclusion of the rule and let Σ[X_1] (and Σ[X_2]) be the premise(s). The rule is applicable only if: Σ[X_1] ⊃ Σ[X_0] and Σ[X_2] ⊑ Σ[X_0].

We conjecture that DBiKt and DBiKt_1 are equivalent and that backward proof search in DBiKt terminates. Note that by equivalence here we mean that DBiKt and DBiKt proves the same set of formulae, but not necessarily the same set of structures. This is because the propagation rules in DBiKt_1 are more restricted so as to allow for easier termination checking. For example, the structure \( A \triangleright \bullet (\triangle A) \) is derivable in DBiKt but not in DBiKt_1, although its formula translate is derivable in both systems. It is likely that a combination of the techniques from [20] and [10] can be used to prove termination of proof search in DBiKt_1, given its similarities to the deep inference systems used in those two works.

6 Semantics

We now give a Kripke-style semantics for BiKt and show that LBiKt is sound with respect to the semantics. Our semantics for BiKt extend Rauszer’s [22] Kripke-style semantics for BiInt by clauses for the tense logic connectives. We use the classical first-order meta-level connectives &,” or”, “not”, ⇒, ∀ and ∃ to state our semantics.

A Kripke frame is a tuple (W, ≤, R_0, R_0) where W is a non-empty set of worlds and ≤ ⊆ (W x W) is a reflexive and transitive binary relation over W, and each of R_0 and R_0 are arbitrary binary relations over W with the following frame conditions:

F1\( \Diamond \) if x ≤ y & xR_0z then ∃w. yR_0w & z ≤ w

F2\( \Box \) if xR_0y & y ≤ z then ∃w. x ≤ w & wR_0z.

A Kripke model extends a Kripke frame with a mapping V from Atoms to 2^W obeying persistence: ∀v ≥ w. w ∈ V(p) ⇒ v ∈ V(p). Given a model (W, ≤, R_0, R_0, V), we say that w ∈ W satisfies p if w ∈ V(p), and write this as w ⊨ p. We write w /⊨ p to mean (not)(w ⊨ p); that is, ∃v ≥ w. v /∈ V(p). The relation ⊨ is then extended to formulae as given in Figure 5. A BiKt-formula A is BiKt-valid if it is satisfied
for every \( w \in W \)
\( w \models \top \)
\( w \models \perp \) for no \( w \in W \)
\( w \models A \land B \) if \( w \models A \) & \( w \models B \)
\( w \models A \lor B \) if \( w \models A \) or \( w \models B \)
\( w \models A \rightarrow B \) if \( \forall v \geq w. v \models A \Rightarrow v \models B \)
\( w \models \neg A \) if \( \forall v \geq w. v \not\models A \)
\( w \models \Diamond A \) if \( \exists v. w R_\Diamond v \land v \models A \)
\( w \models \Box A \) if \( \forall z. v \leq z \land z R_\Box v \Rightarrow v \models A \)
\( w \models \neg \Diamond A \) if \( \exists v. w R_\Diamond v \land v \not\models A \)
\( w \models \neg \Box A \) if \( \exists z. v \leq z \land z R_\Box v \not\Rightarrow v \models A \)

Fig. 5. Semantics for BiKt

by every world in every Kripke model. A nested sequent \( X \triangleright Y \) is BiKt-valid if its formula translation is BiKt-valid.

Our semantics differ from those of Simpson [24] and Ewald [6] because we use two modal accessibility relations instead of one. In our calculi, there is no direct relationship between \( \Diamond \) and \( \Box \) (or \( \Diamond \) and \( \blacksquare \)), but \( \Diamond \) and \( \Box \) are a residuated pair, as are \( \Diamond \) and \( \blacksquare \). Semantically, this corresponds to \( R_\Diamond = R_\Box^{-1} \) and \( R_\blacksquare = R_\Diamond^{-1} \); therefore the clauses in Figure 5 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” [24, page 51] while for us \( F2\Box \) is simply the “persistence of \( \Diamond \)”. 

LBiKt is sound with respect to BiKt. The soundness proof is straightforward by the definition of the semantics and the inference rules.

**Theorem 6.1 (Soundness)** If \( A \) is a BiKt-formula and \( \emptyset \triangleright A \) is LBiKt-derivable, then \( A \) is BiKt-valid.

We conjecture that DBiKt is also complete w.r.t. the semantics; an outline of the proof is given in the extended version of the paper.

7 Modularity, Extensions and Classicality

We first exhibit the modularity of our deep calculus DBiKt by showing that fragments of DBiKt obtained by restricting the language of formulae and structures also satisfy cut admissibility. We then show how we can obtain Ewald’s intuitionistic tense logic IKt [6], Simpson’s intuitionistic modal logic IK [24] 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 [24].

**Modularity**

A nested sequent is purely modal if contains no occurrences of \( \bullet \) nor its formula translates \( \blacksquare \) and \( \Diamond \). 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 \( \Box L_2 \) and \( \Diamond R_2 \). The logical system DBInt is obtained by adding to DInt the deep introduction rules for \( \prec \). In the following, we say that a formula is an IntK-formula if it is composed from propositional variables, intuitionistic connectives, and \( \Box \) and \( \Diamond \). Observe that in DBiKt, the only rules that create \( \bullet \) upwards are \( \Diamond L \) and \( \blacksquare R \). Thus in every DBiKt-derivation \( \pi \) of an IntK formula, the internal sequents in \( \pi \) are purely modal, and hence \( \pi \) is also a DIntK-derivation. This observation gives immediately the following modularity result.
Theorem 7.1 (Modularity) Let $A$ be an Int (resp. BiInt and IntK) formula. The nested sequent $\emptyset \triangleright A$ is DInt-derivable (resp. DBInt- and DIntK-derivable) iff $\emptyset \triangleright A$ is DBiKt-derivable.

A consequence of Theorem 3.5, Theorem 4.1, Theorem 4.7 and Theorem 7.1, is that the cut rule is admissible in DInt, DBInt and DIntK. As the semantics of LBiKt (hence, also DBiKt) is conservative w.r.t. to the semantics of both intuitionistic and bi-intuitionistic logic, the following completeness result holds.

Theorem 7.2 An Int (resp. BiInt) formula $A$ is valid in Int (resp. BiInt) iff $\emptyset \triangleright A$ is derivable in DInt (resp. DBInt).

Obtaining Ewald’s IKt
To obtain Ewald’s IKt [6] we need to collapse $R\Diamond$ and $R\Box$ into one temporal relation $R$ and leave out our semantic clauses for $\prec$ and $\sim$. 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:

$$
\frac{X \triangleright \bullet Y \triangleright \bullet Z}{X \triangleright \bullet (Y \triangleright Z)} \quad \frac{X \triangleright \circ Y \triangleright \circ Z}{X \triangleright \circ (Y \triangleright Z)}
$$

We refer to the extension of LBiKt with these two structural rules as LBiKtE.

Simpson’s intuitionistic modal logic IK [24] can then be obtained from Ewald’s system by restricting the language to the modal fragment. Note that cut-elimination still holds for LBiKtE because these structural rules are closed under formula substitution and the cut-elimination proof for LBiKt still goes through when additional structural rules of this kind are added. We refer the reader to [10] for a discussion on how cut elimination can be proved for this kind of extension.

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 [6]. The rules $\triangleright R$ and $\bullet R$ are sound for $E$-frames. The proofs of the following lemmas can be found in Appendix B.

Lemma 7.3 Rule $\triangleright R$ is sound iff $R\Box \subseteq R\Diamond$.

Lemma 7.4 Rule $\bullet R$ is sound iff $R\Diamond \subseteq R\Box$.

Theorem 7.5 If $A$ is derivable in LBiKtE then $A$ is $E$-valid.

Proof. Straightforward from the soundness of LBiKt w.r.t. BiKt-semantics (which subsumes Ewald’s semantics) and Lemma 7.3 and Lemma 7.4. ⊣

Completeness of LBiKtE w.r.t. IKt and IK can be shown by deriving the axioms of IKt and IK.

Theorem 7.6 System LBiKtE is complete w.r.t. Ewald’s IKt and Simpson’s IK.

Theorem 7.7 (Conservativity over IKt and IK) If $A$ is an IKt-formula (IK formula), then $A$ is IKt-valid (IK-valid) iff $\emptyset \triangleright A$ is derivable in LBiKtE.
Goré, Postniece and Tiu

\[
\begin{array}{l}
\Sigma^-[A, \Box A] \quad T \Box \\
\Sigma^-[\Box A] \quad 4\Box L \\
\frac{p \Box \Box p \rightarrow p}{\Box p \rightarrow p} \quad \text{id} \\
\frac{\Box p \rightarrow \Box p}{\Box p \rightarrow \Box p} \quad \text{id} \\
\frac{\Box p \rightarrow \Box p}{\Box p \rightarrow \Box p} \quad \text{id} \\
\frac{\Box p \rightarrow \Box p}{\Box p \rightarrow \Box p} \quad \text{id} \\
\frac{\Box p \rightarrow \Box p}{\Box p \rightarrow \Box p} \quad \text{id} \\
\end{array}
\]

Regaining classical tense logic \( Kt \)

To collapse BiKt to classical tense logic \( Kt \) we add the rules \( \diamondsuit_R \) and \( \Diamond L \), giving Ewald’s IKt with \( R \Diamond = R \Box \) via Lemmas 7.3-7.4, and then add following two rules:

\[
\begin{array}{l}
\left( X_1 \triangleleft Y_1 \right), X_2 \triangleright Y_2 \quad \left[ X_1 \triangleright Y_1, \left( X_2 \triangleright Y_2 \right) \right] \\
\end{array}
\]

The law of the excluded middle and the law of (dual-)contradiction can then be derived as shown below:

\[
\begin{array}{l}
\left( \Box p \rightarrow \Box p \right) \rightarrow \Box p \quad \left( \Box p \rightarrow \Box p \right) \rightarrow \Box p \\
\end{array}
\]

Further extensions

Our previous work on deep inference systems for classical tense logic [10] shows that extensions of classical tense logic with some standard modal axioms can be formalised by adding numerous propagation rules to the deep inference system for classical tense logic given in that paper. We illustrate here with a few examples how such an approach to extensions with modal axioms can be applied to BiKt. Figure 6 shows the propagation rules that are needed to derive axiom T, 4 and B. For each rule, the derivation of the corresponding axiom is given below the rule. Other nesting combinations will be needed for full completeness. Dual rules allow derivations of \( p \rightarrow \Diamond p \) and \( \Diamond \Diamond p \rightarrow \Diamond p \). The complete treatement of these and other possible extensions of \( LBiKt \) is left for future work.

References

A Admissibility of general contraction

In the following, we label a dashed line with the lemma used to obtain the conclusion from the premise.

Lemma A.1 If \( \vdash_{DBiKt} \pi : \Sigma^+ [\oslash (X \triangleright Y), \oslash Y] \) and contraction on structures is admissible for all derivations \( \pi_1 \) such that \( |\pi_1| \leq |\pi| \) then \( \vdash_{DBiKt} \pi' : \Sigma^+ [\oslash (X \triangleright Y)] \).

Proof. By induction on the height of \( \pi \). The interesting cases are when \( \pi \) ends with a propagation rule that moves a formula into either \( \oslash (X \triangleright Y) \) or \( \oslash Y \):

1. Suppose \( \pi \) ends as below left. Then by Lemma 4.4(ii), there is a derivation \( \pi_2 \) of \( \Box A \triangleright \oslash (A, X \triangleright Y), \oslash Y \) such that \( |\pi_2| \leq |\pi_1| \). Then we can apply the induction hypothesis to \( \pi_2 \) to obtain a derivation \( \pi_3 \) of \( \Box A \triangleright \oslash (A, X \triangleright Y) \). Then the derivation below right gives the required:

2. Suppose \( \pi \) ends as below left. Then applying Lemma 4.2 twice, we obtain a derivation \( \pi_2 \) of \( \Box A \triangleright \oslash (A, X \triangleright Y), \oslash Y \) such that \( |\pi_2| \leq |\pi_1| \). Then we apply the assumption of this lemma to \( \pi_2 \) to obtain a derivation \( \pi_3 \) of \( \Box A \triangleright \oslash (A, X \triangleright Y) \). Then the derivation below right gives the required:

...
Lemma A.2 If $\vdash_{\text{DBiKt}} \pi : \Sigma^-[\circ(X \triangleright Y), \circ X]$ then $\vdash_{\text{DBiKt}} \pi' : \Sigma^-[\circ(X \triangleright Y)]$.

**Proof.** By induction on the height of $\pi$. The interesting cases are when $\pi$ ends with a propagation rule that moves a formula into either $\circ(X \triangleright Y)$ or $\circ X$:

* Suppose $\pi$ ends as below left. Then by Lemma 4.4(i), there is a derivation $\pi_2$ of $\circ(X \triangleright Y, A), \circ X \triangleright \bigtriangledown A$ such that $|\pi_2| \leq |\pi_1|$. Then we can apply the induction hypothesis to $\pi_2$ to obtain a derivation $\pi_3$ of $\circ(X \triangleright Y, A) \triangleright \bigtriangledown A$. Then the derivation below right gives the required:

\[
\begin{array}{c}
\pi_1 \\
\circ((X \triangleright Y) \triangleright A), \circ X \triangleright \bigtriangledown A \quad \frac{}{\circ(X \triangleright Y), \circ X \triangleright \bigtriangledown A} \quad \frac{}{\Diamond R_2}
\end{array}
\quad \pi_3 \\
\quad \frac{}{\circ((X \triangleright Y) \triangleright A) \triangleright \bigtriangledown A} \quad \frac{}{\circ(X \triangleright Y) \triangleright \bigtriangledown A} \quad \frac{}{\Diamond R_2}
\]

* Suppose $\pi$ ends as below left. Then applying Lemma 4.2 twice, we obtain a derivation $\pi_2$ of $\circ(X \triangleright Y, A), \circ(X \triangleright Y, A) \triangleright \bigtriangledown A$ such that $|\pi_2| \leq |\pi_1|$. Then we apply the assumption of this lemma to $\pi_2$ to obtain a derivation $\pi_3$ of $\circ(X \triangleright Y, A) \triangleright \bigtriangledown A$. Then the derivation below right gives the required:

\[
\begin{array}{c}
\pi_1 \\
\circ(X \triangleright Y), \circ(X \triangleright A) \triangleright \bigtriangledown A \quad \frac{}{\circ(X \triangleright Y), \circ X \triangleright \bigtriangledown A} \quad \frac{}{\Diamond R_2}
\end{array}
\quad \pi_3 \\
\quad \frac{}{\circ((X \triangleright Y) \triangleright A) \triangleright \bigtriangledown A} \quad \frac{}{\circ(X \triangleright Y) \triangleright \bigtriangledown A} \quad \frac{}{\Diamond R_2}
\]

* For the case where $Y$ is a $\triangleright$-structure and $\pi$ ends with a propagation rule applied to $Y$, there are three non-trivial sub-cases:

- For the base case, use Lemma 4.3.
- For the case where $Y$ is a $\triangleright$-structure, we show the sub-case where $Y$ in a negative context, the other case is symmetric:

\[
\begin{array}{c}
\Sigma[(Y_1 \triangleright Y_2), (Y_1 \triangleright Y_2) \triangleright Z] \\
\Sigma[(Y_1 \triangleright Y_2, Y_1 \triangleright Y_2) \triangleright Z] \\
\Sigma[(Y_1 \triangleright Y_2) \triangleright Z] \\
\Sigma[(Y_1 \triangleright Y_2) \triangleright Z] \\
\Sigma[(Y_1 \triangleright Y_2) \triangleright Z] \\
\Sigma[(Y_1 \triangleright Y_2) \triangleright Z]
\end{array}
\]

\[
\begin{array}{c}
\text{Lemma 4.4(i)} \\
\text{Lemma 4.4(i)} \\
\text{IH} \\
\text{IH} \\
\Sigma[(Y_1 \triangleright Y_2) \triangleright Z] \\
\text{Lemma 4.4(iii)}
\end{array}
\]

- For the case where $Y$ is a $\circ-$ or $\bullet-$structure and $\pi$ ends with a propagation rule applied to $Y$, there are three non-trivial sub-cases:

- For the case where $Y$ is a $\circ-$ or $\bullet-$structure and $\pi$ ends with a propagation rule applied to $Y$, there are three non-trivial sub-cases:
Fig. B.1. Derivations of Simpson’s axiom 2 and Ewald’s axiom 5 (left) and Ewald’s axiom 7 (right)

- A formula is propagated into Y and Y is in a positive context, as below left. Then by Lemma A.1, there is a derivation \( \pi_1' \) of \( \Box A, X \triangleright o(A \triangleright Z) \). Then the derivation below right gives the required:

\[
\begin{array}{c}
\Box A, X \triangleright o(A \triangleright Z), oZ \\hline
\Box A, X \triangleright oZ, oZ \quad \Box L2
\end{array}
\]

\[
\begin{array}{c}
\Box A, X \triangleright oZ, oZ \\hline
\Box A, X \triangleright oZ \quad \Box L2
\end{array}
\]

- A formula is propagated into Y and Y is in a negative context, as below left. Then by Lemma A.2, there is a derivation \( \pi_1' \) of \( o(Z \triangleright A) \triangleright X, \Diamond A \). Then the derivation below right gives the required:

\[
\begin{array}{c}
o(Z \triangleright A), oZ \triangleright X, \Diamond A \\hline
o(Z \triangleright A), oZ \triangleright X, \Diamond A \quad \Diamond R2
\end{array}
\]

\[
\begin{array}{c}
o(Z \triangleright A) \triangleright X, \Diamond A \\hline
o(Z \triangleright A) \triangleright X, \Diamond A \quad \Diamond R2
\end{array}
\]

- A formula is propagated out of Y, as below left. In this case we use the sub-induction hypothesis to obtain a derivation \( \pi_1' \) of \( X \triangleright A, o(\Diamond A, Z) \). Then the derivation below right gives the required:

\[
\begin{array}{c}
X \triangleright A, o(\Diamond A, Z), o(\Diamond A, Z) \\hline
X \triangleright o(\Diamond A, Z) \quad \Diamond R1
\end{array}
\]

\[
\begin{array}{c}
X \triangleright o(\Diamond A, Z) \\hline
X \triangleright o(\Diamond A, Z) \quad \Diamond R1
\end{array}
\]

B Modularity, Extensions and Classicality

**Theorem 7.6.** System LB\textbf{i}Kt\textbf{E} is complete w.r.t. Ewald’s IKt and Simpson’s IK.

**Proof.** We show the non-trivial cases; the rest are similar or easier. Derivations of Simpson’s axiom 2 and Ewald’s axiom 5 and 7 are given in Figure B.1, derivations of Simpson’s axiom 5 and Ewald’s axiom 10 and 11’ are given in Figure B.2. \( \end{proof} \)
Lemma 7.3. Rule \( \Diamond \Box \) 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 \( \Diamond A \rightarrow \Box B \) of the premise is valid. We then show that the formula translation \( \Box(A \rightarrow B) \) of the conclusion is valid. For a contradiction, suppose that \( \Box(A \rightarrow B) \) is not valid. That is, there exists a world \( u \) such that \( u \not\models \Box(p \rightarrow q) \). Then (4) there exist worlds \( x \) and \( y \) such that \( u \models x \& xR\Box y \) and \( y \not\models p \rightarrow q \). Thus there exists \( z \) s.t. \( z \models y \) and \( z \models \Diamond(p \rightarrow q) \). 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 \( wR_\Box z \) too, meaning that \( w \models \Diamond p \). From (2) we get \( w \models \Box q \), which gives us \( z \models q \), giving us the contradiction we seek. Therefore the premise \( \Box(A \rightarrow B) \) 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 rule implies that \( \Box(\Diamond A \rightarrow \Box B) \rightarrow \Box(A \rightarrow B) \) is derivable. 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 \( xR_\Diamond y \). Let \( W = \{u, w, x, y, z\} \), let \( < \) be the relation \( \{(u, x), (x, w), (y, z)\} \) and let \( \leq \) be the reflexive-transitive closure of \( < \). Let \( R_\Diamond = \{\} \), \( R_\Box = \{(x, y), (w, z)\} \) and let \( V(p) = \{z\}, V(q) = \{\} \). Then the model \( (W, \leq, R_\Diamond, R_\Box, V) \) satisfies (5), and has \( u \models \Diamond p \rightarrow \Box q \) but \( u \not\models \Box(p \rightarrow q) \). \( \Rightarrow \)

Lemma 7.4. Rule \( \Diamond \Box \) is sound iff \( R_\Box \subseteq R_\Diamond \).

Proof. \( R_\Box \subseteq R_\Box \) means \( R_\Box \subseteq R_\diamond \); the rest of the proof is analogous to the proof of Lemma 7.3. \( \Rightarrow \)