Cut-elimination and Proof-search for Bi-Intuitionistic Logic Using Nested Sequents

Rajeev Goré    Linda Postniece    Alwen Tiu

Computer Sciences Laboratory
The Australian National University

Advances in Modal Logic 2008
Bi-Intuitionistic Logic

- Int + dual-Int
Bi-Intuitionistic Logic

- \( \text{Int} + \text{dual-Int} \)

- \( \prec \) dual to \( \rightarrow \)

\[
\frac{A \vdash B, \Delta}{A \prec B \vdash \Delta} \quad \prec_L \quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \quad \rightarrow_R
\]
Bi-Intuitionistic Logic

- Int + dual-Int
- $\vdash$ dual to $\rightarrow$
  \[
  \frac{A \vdash B, \Delta}{A \vdash B \vdash \Delta} \quad \vdash_L
  \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \quad \vdash_R
  \]
- Hilbert calculus, algebraic and Kripke semantics (Rauszer 1974)
Bi-Intuitionistic Logic

- Int + dual-Int
- $\vdash A \vdash B$, $\Delta$  \[ \frac{A \vdash B}{A \vdash B \vdash \Delta} \quad \vdash_L \quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \quad \vdash_R$
- Hilbert calculus, algebraic and Kripke semantics (Rauszer 1974)
- Type theoretic interpretation of co-routines (Crolard 2004)
Bi-Intuitionistic Logic

- Int + dual-Int
- $\vdash A \dashv B, \Delta \quad \vdash A \dashv B \dashv \Delta \\
  \vdash A \dashv B \vdash \Delta \\
- \vdash \Gamma, A \vdash B \\
  \vdash \Gamma \vdash A \dashv B \\
- \vdash \Gamma, A \vdash B \\
  \vdash \Gamma \vdash A \dashv B \\
- \vdash \Gamma, A \vdash B \\
  \vdash \Gamma \vdash A \dashv B \\
- \vdash \Gamma, A \vdash B \\
  \vdash \Gamma \vdash A \dashv B \\

- Hilbert calculus, algebraic and Kripke semantics (Rauszer 1974)
- Type theoretic interpretation of co-routines (Crolard 2004)
- "Cut-free" sequent calculus (Rauszer 1974)
Bi-Intuitionistic Logic

- Int + dual-Int
  - $\frac{A \vdash B, \Delta}{A \dashv B \vdash \Delta}$\hspace{1cm} $\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B}$

- Hilbert calculus, algebraic and Kripke semantics (Rauszer 1974)
- Type theoretic interpretation of co-routines (Crolard 2004)
- “Cut-free” sequent calculus (Rauszer 1974)
- Display calculus with cut-elimination (Goré 1998)
Motivation and Related Work

• Rauszer’s cut-elimination fails (Uustalu 2006)
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
Motivation and Related Work

• Rauszer’s cut-elimination fails (Uustalu 2006)
  • Some proofs require cut rule (bad for proof-search)
• Crolard’s calculus uses cuts (Crolard 2001)
• Goré’s display calculus is not suitable for proof-search
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
- GBiInt semantic completeness (Buisman/Postniece, Goré 2007)
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
- GBiInt semantic completeness (Buisman/Postniece, Goré 2007)
- Goal: a sequent calculus for Bilnt that
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
- GBiInt semantic completeness (Buisman/Postniece, Goré 2007)
- Goal: a sequent calculus for Bilnt that
  - Is complete
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
- GBiInt semantic completeness (Buisman/Postniece, Goré 2007)
- Goal: a sequent calculus for Bilnt that
  - Is complete
  - Allows backward proof search
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
- GBiInt semantic completeness (Buisman/Postniece, Goré 2007)
- Goal: a sequent calculus for Bilnt that
  - Is complete
  - Allows backward proof search
  - Has syntactic cut-elimination
Motivation and Related Work

- Rauszer’s cut-elimination fails (Uustalu 2006)
  - Some proofs require cut rule (bad for proof-search)
- Crolard’s calculus uses cuts (Crolard 2001)
- Goré’s display calculus is not suitable for proof-search
  - Unrestricted display postulates
  - Unrestricted general contraction
- Labelled sequent calculus (Uustalu, Pinto 2006, no details)
- GBiInt semantic completeness (Buisman/Postniece, Goré 2007)
- Goal: a sequent calculus for BiInt that
  - Is complete
  - Allows backward proof search
  - Has syntactic cut-elimination
- Goal: proof search in display calculus
1. Bi-Intuitionistic Logic
   - Syntax and Semantics
   - Bilnt Challenges

2. Nested Sequents
   - Structures
   - LBilnt₁

3. Cut-Elimination
   - Atomic Cuts
   - General Cuts

4. Proof Search
   - LBilnt₂
   - Strategy
   - Termination

5. Conclusion
• **Connectives:** $\land \lor \to \leftarrow$
Syntax and Kripke Models

- **Connectives:** $\land \lor \to \leftarrow$
- **Constants:** $\top \bot$
Syntax and Kripke Models

- **Connectives:** \( \land, \lor, \rightarrow, \leftarrow \)
- **Constants:** \( \top, \bot \)
- **Defined connectives:** \( \neg, \sim \)
Syntax and Kripke Models

- **Connectives:** $\land \lor \rightarrow <$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \rightarrow \bot$ (Int negation)
Syntax and Kripke Models

- **Connectives:** $\land \lor \to \leftarrow$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \to \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
Syntax and Kripke Models

- **Connectives:** $\land \lor \to \leftarrow$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \to \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics:** model $\langle W, \leq, V \rangle$
Syntax and Kripke Models

- **Connectives**: $\land \lor \rightarrow \leftarrow$
- **Constants**: $\top \bot$
- **Defined connectives**: $\neg \sim$
  - $\neg A := A \rightarrow \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics**: model $\langle W, \leq, V \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
Syntax and Kripke Models

- **Connectives:** $\land, \lor, \rightarrow, \leftarrow$
- **Constants:** $\top, \bot$
- **Defined connectives:** $\neg, \sim$
  - $\neg A := A \rightarrow \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics:** model $\langle W, \leq, V \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $V$ maps atoms to $2^W$
Syntax and Kripke Models

- **Connectives:** \( \land, \lor, \rightarrow, \leftarrow \)
- **Constants:** \( \top, \bot \)
- **Defined connectives:** \( \neg, \sim \)
  - \( \neg A := A \rightarrow \bot \) (Int negation)
  - \( \sim A := \top \leftarrow A \) (dual-Int / paraconsistent negation)
- **Kripke semantics:** model \( \langle W, \leq, V \rangle \)
  - \( \leq \) is a reflexive and transitive binary relation over \( W \)
  - \( V \) maps atoms to \( 2^W \)
  - \( V(\top) = W \) and \( V(\bot) = \emptyset \)
Syntax and Kripke Models

- **Connectives:** $\land \lor \to \leftarrow$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \to \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics:** model $\langle W, \leq, V \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $V$ maps atoms to $2^W$
  - $V(\top) = W$ and $V(\bot) = \emptyset$
  - $V$ satisfies: if $w \in V(p)$ and $w \leq u$ then $u \in V(p)$
Syntax and Kripke Models

- **Connectives:** $\land \lor \to \prec$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \to \bot$ (Int negation)
  - $\sim A := \top \prec A$ (dual-Int / paraconsistent negation)
- **Kripke semantics:** model $\langle W, \leq, V \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $V$ maps atoms to $2^W$
  - $V(\top) = W$ and $V(\bot) = \emptyset$
  - $V$ satisfies: if $w \in V(p)$ and $w \leq u$ then $u \in V(p)$
  - $w \models p$ iff $w \in V(p)$
Syntax and Kripke Models

- **Connectives:** $\land \lor \rightarrow \leftarrow$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \rightarrow \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics:** model $\langle W, \leq, \mathcal{V} \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $\mathcal{V}$ maps atoms to $2^W$
  - $\mathcal{V}(\top) = W$ and $\mathcal{V}(\bot) = \emptyset$
  - $\mathcal{V}$ satisfies: if $w \in \mathcal{V}(p)$ and $w \leq u$ then $u \in \mathcal{V}(p)$
  - $w \models p$ iff $w \in \mathcal{V}(p)$
  - $w \models A \land B$ iff $w \models A$ & $w \models B$
Syntax and Kripke Models

- **Connectives:** $\land \lor \rightarrow \leftarrow$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \rightarrow \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics: model** $\langle W, \leq, V \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $V$ maps atoms to $2^W$
  - $V(\top) = W$ and $V(\bot) = \emptyset$
  - $V$ satisfies: if $w \in V(p)$ and $w \leq u$ then $u \in V(p)$
  - $w \models p$ iff $w \in V(p)$
  - $w \models A \land B$ iff $w \models A$ and $w \models B$
  - $w \models A \lor B$ iff $w \models A$ or $w \models B$
Syntax and Kripke Models

- **Connectives:** $\land \lor \rightarrow \leftarrow$
- **Constants:** $\top \bot$
- **Defined connectives:** $\neg \sim$
  - $\neg A := A \rightarrow \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics: model $\langle W, \leq, V \rangle$**
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $V$ maps atoms to $2^W$
  - $V(\top) = W$ and $V(\bot) = \emptyset$
  - $V$ satisfies: if $w \in V(p)$ and $w \leq u$ then $u \in V(p)$
  - $w \models p$ iff $w \in V(p)$
  - $w \models A \land B$ iff $w \models A$ and $w \models B$
  - $w \models A \lor B$ iff $w \models A$ or $w \models B$
  - $w \models A \rightarrow B$ iff $\forall u \geq w. u \models A$ or $u \models B$
Syntax and Semantics

- **Connectives**: $\land \lor \to \leftarrow$
- **Constants**: $\top \bot$
- **Defined connectives**: $\neg \sim$
  - $\neg A := A \to \bot$ (Int negation)
  - $\sim A := \top \leftarrow A$ (dual-Int / paraconsistent negation)
- **Kripke semantics**: model $\langle W, \leq, V \rangle$
  - $\leq$ is a reflexive and transitive binary relation over $W$
  - $V$ maps atoms to $2^W$
  - $V(\top) = W$ and $V(\bot) = \emptyset$
  - $V$ satisfies: if $w \in V(p)$ and $w \leq u$ then $u \in V(p)$
  - $w \Vdash p$ iff $w \in V(p)$
  - $w \Vdash A \land B$ iff $w \Vdash A$ \& $w \Vdash B$
  - $w \Vdash A \lor B$ iff $w \Vdash A$ or $w \Vdash B$
  - $w \Vdash A \rightarrow B$ iff $\forall u \geq w. u \nvdash A$ or $u \Vdash B$
  - $w \Vdash A \leftarrow B$ iff $\exists u \leq w. u \Vdash A$ \& $u \nvdash B$
Uustalu’s Example: Using Cut

- Rauszer’s $\leftarrow^L$ and $\rightarrow^R$ require singleton antecedent/succedent:
Uustalu’s Example: Using Cut

- Rauszer’s $\prec_L$ and $\to_R$ require singleton antecedent/succedent:

$$\frac{A \vdash B, \Delta}{A \prec B \vdash \Delta} \quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$
Uustalu’s Example: Using Cut

- Rauszer’s $\langle L \rangle$ and $\rightarrow_R$ require singleton antecedent/succedent:
  \[
  \frac{A \vdash B, \Delta}{A \langle L \rangle B \vdash \Delta} \quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \rightarrow_R
  \]
- $p \vdash q, r \rightarrow ((p \langle L \rangle q) \land r)$ is not cut-free derivable in Rauszer’s G1
Uustalu’s Example: Using Cut

- Rauszer’s $\langle L \rangle$ and $\rightarrow_R$ require singleton antecedent/succedent:
  - $A \vdash B, \Delta \quad A \langle B \vdash \Delta \quad \langle L \rangle$
  - $\Gamma, A \vdash B \quad \Gamma \vdash A \rightarrow B \quad \rightarrow_R$
- $p \vdash q, r \rightarrow ((p \langle q) \wedge r)$ is not cut-free derivable in Rauszer’s G1
- Derivation using cut:
Uustalu’s Example: Using Cut

- Rauszer’s \( \langle_L \) and \( \rightarrow_R \) require singleton antecedent/succedent:
  \[
  \frac{A \vdash B, \Delta}{A \langle L \rangle B \vdash \Delta} \quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \rightarrow_R
  \]

- \( p \vdash q, r \rightarrow ((p \langle q \rangle) \wedge r) \) is not cut-free derivable in Rauszer’s G1

- Derivation using cut:
  \[
  \frac{p \vdash q, \ld \quad q \vdash q, \ld}{\frac{p \vdash q, p \quad q \vdash q}{p \vdash q, p \langle q \rangle} \langle L \rangle} \quad \frac{p \vdash q, r \vdash p \langle q \rangle, \ld}{\frac{p \vdash q, r \vdash p \langle q \rangle, q}{\frac{p \vdash q, r \vdash p \langle q \rangle, r}{\frac{p \vdash q, r \vdash p \langle q \rangle}{\frac{p \vdash q, r \vdash p \langle q \rangle, \wedge R}{\frac{p \vdash q, r \vdash (p \langle q \rangle) \wedge r}{\frac{p \vdash q, r \vdash ((p \langle q \rangle) \wedge r)}{\rightarrow R}}}}}\]
\]

\( p \vdash q, r \rightarrow ((p \langle q \rangle) \wedge r) \)
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
Nested Sequents

- Negative structures: \( N := \emptyset \mid A \mid (N, N) \mid N < P \)
- Positive structures: \( P := \emptyset \mid A \mid (P, P) \mid N > P \)
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
- Positive structures: $P := \emptyset \mid A \mid (P, P) \mid N > P$
- Sequents: $X \vdash Y$ where
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
- Positive structures: $P := \emptyset \mid A \mid (P, P) \mid N > P$
- Sequents: $X \vdash Y$ where
  - $X$ is a negative structure
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
- Positive structures: $P := \emptyset \mid A \mid (P, P) \mid N > P$
- Sequents: $X \vdash Y$ where
  - $X$ is a negative structure
  - $Y$ is a positive structure
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
- Positive structures: $P := \emptyset \mid A \mid (P, P) \mid N > P$
- Sequents: $X \vdash Y$ where
  - $X$ is a negative structure
  - $Y$ is a positive structure
- Similar, but more restricted than display logic
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
- Positive structures: $P := \emptyset \mid A \mid (P, P) \mid N > P$
- Sequents: $X \vdash Y$ where
  - $X$ is a negative structure
  - $Y$ is a positive structure
- Similar, but more restricted than display logic
- Examples:
Nested Sequents

- Negative structures: $N := \emptyset \mid A \mid (N, N) \mid N < P$
- Positive structures: $P := \emptyset \mid A \mid (P, P) \mid N > P$
- Sequents: $X \vdash Y$ where
  - $X$ is a negative structure
  - $Y$ is a positive structure
- Similar, but more restricted than display logic
- Examples:
  - $r \vdash p \leftarrow q$
Nested Sequents

- **Negative structures:** $N := \emptyset \mid A \mid (N, N) \mid N < P$
- **Positive structures:** $P := \emptyset \mid A \mid (P, P) \mid N > P$
- **Sequents:** $X \vdash Y$ where
  - $X$ is a negative structure
  - $Y$ is a positive structure
- **Similar, but more restricted than display logic**
- **Examples:**
  - $r \vdash p \prec q$
  - $(p \prec q), r \vdash (q > r), ((p < q) > w)$
Identity, cut and structural rules

Identity and cut:

\[ X, A \vdash A, Y \quad \text{id} \quad \frac{X_1 \vdash Y_1, A \quad A, X_2 \vdash Y_2}{X_1, X_2 \vdash Y_1, Y_2} \quad \text{cut} \]
Identity, cut and structural rules

Identity and cut:

\[
\begin{align*}
&\text{id} & & X, A \vdash A, Y \\
&\text{cut} & & X_1 \vdash Y_1, A, X_2 \vdash Y_2 \quad X_1, X_2 \vdash Y_1, Y_2
\end{align*}
\]

Structural rules:

\[
\begin{align*}
&\text{w}_L & & X \vdash Y \quad X, A \vdash Y \\
&\text{w}_R & & X \vdash Y \quad X, A \vdash Y \\
&\text{c}_L & & X, A, A \vdash Y \\
&\text{c}_R & & X \vdash A, A, Y
\end{align*}
\]
Identity, cut and structural rules

Identity and cut:

Identity and cut:

\[ X, A \vdash A, Y \quad \text{id} \]

\[ X_1 \vdash Y_1, A, X_2 \vdash Y_2 \quad \text{cut} \]

Structural rules:

Structural rules:

\[ \frac{X \vdash Y}{X, A \vdash Y} \quad w_L \]

\[ \frac{X \vdash Y}{X \vdash A, Y} \quad w_R \]

\[ \frac{X, A, A \vdash Y}{X, A \vdash Y} \quad c_L \]

\[ \frac{X \vdash A, A, Y}{X \vdash A, Y} \quad c_R \]
Identity, cut and structural rules

Identity and cut:

\[ \frac{X, A \vdash A, Y}{X, A \vdash A, Y} \quad id \quad \frac{X_1 \vdash Y_1, A \quad A, X_2 \vdash Y_2}{X_1, X_2 \vdash Y_1, Y_2} \quad cut \]

Structural rules:

\[ \frac{X \vdash Y}{X, A \vdash Y} \quad w_L \quad \frac{X \vdash Y}{X \vdash A, Y} \quad w_R \]

\[ \frac{X, A, A \vdash Y}{X, A \vdash A, Y} \quad c_L \quad \frac{X \vdash A, A, Y}{X \vdash A, A, Y} \quad c_R \]

\[ \frac{X_1 < Y_1, X_2 \vdash Y_2}{X_1, X_2 \vdash Y_1, Y_2} \quad s_L \quad \frac{X_1 \vdash Y_1, (X_2 > Y_2)}{X_1, X_2 \vdash Y_1, Y_2} \quad s_R \]

\[ \frac{X_2 \vdash Y_2, Y_1}{X_1, (X_2 < Y_2) \vdash Y_1} < \quad \frac{X_1, X_2 \vdash Y_2}{X_1 \vdash Y_1, (X_2 > Y_2)} > \]
Logical rules

\[
\frac{X \vdash A, Y \quad X, B \vdash Y}{X, A \rightarrow B \vdash Y} \quad \rightarrow_L \quad \frac{X, A \vdash B}{X \vdash Y, A \rightarrow B} \quad \rightarrow_R
\]
Logical rules

\[
\frac{X \vdash A, Y \quad X, B \vdash Y}{X, A \rightarrow B \vdash Y} \quad \rightarrow_L \quad \frac{X, A \vdash B}{X \vdash Y, A \rightarrow B} \quad \rightarrow_R
\]
Logical rules

\[ \frac{X \vdash A, Y \quad X, B \vdash Y}{X, A \to B \vdash Y} \rightarrow_L \quad \frac{X, A \vdash B}{X \vdash Y, A \to B} \rightarrow_R \]

\[ \frac{A \vdash B, Y}{X, A \angle B \vdash Y} \angle_L \quad \frac{X \vdash A, Y \quad X, B \vdash Y}{X \vdash A \angle B, Y} \angle_R \]
Uustalu’s Example Revisited

Using cut:

\[\frac{p \vdash q, p}{\vdash q, p \vdash q} \quad \text{Id} \quad \frac{q \vdash q}{\vdash q, p \vdash q} \quad \text{Id} \quad \frac{p \vdash q, r \vdash p \vdash q}{ld} \quad \frac{p \vdash q, r \vdash r}{ld} \quad \frac{p \vdash q, r}{\vdash q \rightarrow ((p \vdash q) \land r)} \quad \text{cut} \]

Using \text{LBiInt}_{1} without cut:

\[\frac{p \vdash q, p}{\vdash q, p \vdash q} \quad \text{Id} \quad \frac{p, q \vdash q}{\vdash q, p \vdash q} \quad \text{Id} \quad \frac{(p \vdash q), r \vdash p \vdash q}{\vdash (p \vdash q), r \vdash (p \vdash q) \land r} \quad \text{Id} \]

\[\frac{p \vdash q, r \vdash (p \vdash q) \land r}{\vdash (p \vdash q), r \rightarrow ((p \vdash q) \land r)} \quad \text{R} \quad \frac{(p \vdash q), r \vdash r}{\vdash (p \vdash q), r \vdash r} \quad \text{Id} \]

\[\frac{(p \vdash q), r \vdash r}{\vdash (p \vdash q), r \vdash (p \vdash q) \land r} \quad \text{Id} \quad \frac{(p \vdash q), r \vdash (p \vdash q) \land r}{\vdash (p \vdash q), r \rightarrow ((p \vdash q) \land r)} \quad \text{R} \]

\[\frac{(p \vdash q), r \vdash (p \vdash q) \land r}{\vdash (p \vdash q), r \rightarrow ((p \vdash q) \land r)} \quad \text{S}_{L} \]

\[\frac{(p \vdash q), r \vdash (p \vdash q) \land r}{\vdash (p \vdash q), r \rightarrow ((p \vdash q) \land r)} \quad \text{R} \]
Lemma

Contraction and weakening on structures admissible:

\[
\frac{X, Y, Y \vdash Z}{X, Y \vdash Z} \quad gc_L
\]

\[
\frac{X \vdash Y, Y, Z}{X \vdash Y, Z} \quad gc_R
\]

\[
\frac{X \vdash Z}{X, Y \vdash Z} \quad gw_L
\]

\[
\frac{X \vdash Z}{X \vdash Y, Z} \quad gw_R
\]
General Contraction and Weakening

Lemma

Contraction and weakening on structures admissible:

\[
\frac{X, Y, Y \vdash Z}{X, Y \vdash Z} \quad \text{gc}_L \quad \frac{X \vdash Y, Y, Z}{X \vdash Y, Z} \quad \text{gc}_R
\]

\[
\frac{X \vdash Z}{X, Y \vdash Z} \quad \text{gw}_L \quad \frac{X \vdash Z}{X \vdash Y, Z} \quad \text{gw}_R
\]

Proof.

By induction on the size of $Y$. 

Rajeev Goré, Linda Postniece, Alwen Tiu

Bi-Intuitionistic Logic
We transform

\[
\frac{p \vdash p \text{ id} \quad \ldots \quad p \vdash p \text{ id}}{\vdash \theta}
\]

\[
\frac{X_1 \vdash Y_1, p}{\pi}
\]

\[
\frac{X_1, X_2 \vdash Y_1, Y_2}{\text{cut}}
\]
Atomic Cuts

We transform

\[
\frac{p \vdash p}{\text{id}} \ldots \frac{p \vdash p}{\text{id}} \\
\vdash \theta \\
\frac{X_1 \vdash Y_1, p}{\text{cut}} \\
\frac{p, X_2 \vdash Y_2}{\pi}
\]

into:

\[
\frac{p, X_2 \vdash Y_2}{\pi} \quad > \quad \ldots \quad > \\
\frac{p \vdash X_2 > Y_2}{\pi} \\
\vdash \theta[p/ X_2 > Y_2] \\
\frac{X_1 \vdash Y_1, (X_2 > Y_2)}{\text{cut}_{SR}}
\]
We transform

\[ \pi_1 \]
\[ X_1', A \vdash B \]
\[ X'_1 \vdash Y'_1, A \rightarrow B \]
\[ \vdots \theta_1 \]
\[ X_1 \vdash Y_1, A \rightarrow B \]
\[ \rightarrow_R \]

\[ \pi_2 \]
\[ X_2' \vdash A, Y'_2 \]
\[ B, X'_2 \vdash Y'_2 \]
\[ A \rightarrow B, X'_2 \vdash Y'_2 \]
\[ \vdots \theta_2 \]
\[ A \rightarrow B, X_2 \vdash Y_2 \]
\[ \rightarrow_L \]

\[ \pi_3 \]
\[ X_1, X_2 \vdash Y_1, Y_2 \]

\[ \text{cut} \]
General Cuts: \( A \rightarrow B \)

We transform

\[
\begin{align*}
\pi_1 & \quad X_1', A \vdash B & \quad \pi_2 & \quad X_2 \vdash A, Y_2' & \quad \pi_3 & \quad B, X_2' \vdash Y_2' \\
X_1' \vdash Y_1', A \rightarrow B & \quad \rightarrow_R & \quad A \rightarrow B, X_2' \vdash Y_2' & \quad \rightarrow_L
\end{align*}
\]

\[\vdash \theta_1 X_1 \vdash Y_1, A \rightarrow B\]

\[\vdash \theta_2 A \rightarrow B, X_2' \vdash Y_2\]

\[\vdash \text{cut}\]

into:

\[
\begin{align*}
\pi_2 & \quad X_2' \vdash A, Y_2' & \quad \pi_1 & \quad X_1', A \vdash B & \quad \pi_3 & \quad B, X_2' \vdash Y_2' \\
X_1', X_2' \vdash Y_2' & \quad \text{cut} & \quad X_1', A, X_2' \vdash Y_2' & \quad \text{cut} & \quad \text{cut}
\end{align*}
\]

\[\vdash \theta_2[A \rightarrow B/X_1']\]

\[\vdash \theta_1[A \rightarrow B/X_2 > Y_2]\]

\[\vdash X_1, X_2 \vdash Y_2\]

\[\vdash X_1', (X_2 > Y_2) \]

\[\vdash X_1, (X_2 > Y_2) \]

\[\vdash X_1, X_2 \vdash Y_1, Y_2 \]

\[\text{gc}_L, \text{gc}_R\]

\[\text{S}_R\]
From \( \text{LBiInt}_1 \) to \( \text{LBiInt}_2 \)

- \( \text{LBiInt}_1 \) has an elegant direct cut-elimination proof
From $\text{LBilInt}_1$ to $\text{LBilInt}_2$

- $\text{LBilInt}_1$ has an elegant direct cut-elimination proof
  - Using structural rules $s_L$, $s_R$, $\succ$, and $\prec$
From $\text{LBilInt}_1$ to $\text{LBilInt}_2$

- $\text{LBilInt}_1$ has an elegant direct cut-elimination proof
  - Using structural rules $s_L$, $s_R$, $>$ and $<$
  - Also possible via detour through display calculus
From $\text{LBiInt}_1$ to $\text{LBiInt}_2$

- $\text{LBiInt}_1$ has an elegant direct cut-elimination proof
  - Using structural rules $s_L$, $s_R$, $>$ and $<$
  - Also possible via detour through display calculus
- But $\text{LBiInt}_1$ is not suitable for proof search:
From $\text{LBilInt}_1$ to $\text{LBilInt}_2$

- $\text{LBilInt}_1$ has an elegant direct cut-elimination proof
  - Using structural rules $s_L$, $s_R$, $>$ and $<$
  - Also possible via detour through display calculus
- But $\text{LBilInt}_1$ is not suitable for proof search:
  - Structural rules allow shuffling of structures ad infinitum
From $\text{LBilInt}_1$ to $\text{LBilInt}_2$

- $\text{LBilInt}_1$ has an elegant direct cut-elimination proof
  - Using structural rules $s_L$, $s_R$, $\triangleright$ and $\triangleleft$
  - Also possible via detour through display calculus
- But $\text{LBilInt}_1$ is not suitable for proof search:
  - Structural rules allow shuffling of structures ad infinitum
  - Unlimited contraction
From $\text{LBilInt}_1$ to $\text{LBilInt}_2$

- $\text{LBilInt}_1$ has an elegant direct cut-elimination proof
  - Using structural rules $s_L$, $s_R$, $>$ and $<$
  - Also possible via detour through display calculus
- But $\text{LBilInt}_1$ is not suitable for proof search:
  - Structural rules allow shuffling of structures ad infinitum
  - Unlimited contraction
- Solution: absorb structural rules into logical rules
From LBiInt\textsubscript{1} to LBiInt\textsubscript{2}

- LBiInt\textsubscript{1} has an elegant direct cut-elimination proof
  - Using structural rules $s\textsubscript{L}$, $s\textsubscript{R}$, $>$ and $<$
  - Also possible via detour through display calculus
- But LBiInt\textsubscript{1} is not suitable for proof search:
  - Structural rules allow shuffling of structures ad infinitum
  - Unlimited contraction
- Solution: absorb structural rules into logical rules

\[
\frac{(X < Y, A \to B), X, A \vdash B}{(X < Y, A \to B), X \vdash Y, A \to B} \quad \rightarrow \quad \frac{(X < Y, A \to B), X \vdash Y, A \to B}{X, X \vdash Y, A \to B, A \to B} \quad \rightarrow \quad \frac{(X < Y, A \to B), \{X\}, A \vdash B}{X \vdash Y, A \to B} \quad \rightarrow\]

$$\{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}$$
LBiInt$_2$ Rules

\( \{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\} \)
LB\textit{Int}_2 Rules

\[
\{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}
\]

\[
X, A \vdash A, Y \quad id
\]
**LBInt₂ Rules**

\[
\{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}
\]

\[
\frac{X, A \vdash A, Y}{id}
\]

\[
\frac{X_2 \vdash Y_2, \{Y_1\}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{Y_1\} \not\subseteq \{Y_2\}
\]

\[
\frac{\{X_1\}, X_2 \vdash Y_2}{X_1 \vdash Y_1, (X_2 > Y_2)} > \{X_1\} \not\subseteq \{X_2\}
\]
LBInt_2 Rules

\[ \{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\} \]

\[ X, A \vdash A, Y \quad id \]

\[ \frac{X_2 \vdash Y_2, \{Y_1\}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{Y_1\} \not\subseteq \{Y_2\} \]

\[ \frac{\{X_1\}, X_2 \vdash Y_2}{X_1 \vdash Y_1, (X_2 > Y_2)} > \{X_1\} \not\subseteq \{X_2\} \]

\[ X, A \rightarrow B \vdash A, Y \quad X, A \rightarrow B, B \vdash Y \]

\[ \rightarrow_L \]

\[ X \vdash Y, A \rightarrow B, B \]

\[ \rightarrow_{R1} \]
\[\{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}\]

\[\frac{X, A \vdash A, Y}{\text{id}}\]

\[\frac{X_2 \vdash Y_2, \{Y_1\}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{Y_1\} \not\subseteq \{Y_2\}\]

\[\frac{\{X_1\}, X_2 \vdash Y_2}{X_1 \vdash Y_1, (X_2 > Y_2)} > \{X_1\} \not\subseteq \{X_2\}\]

\[\frac{X, A \rightarrow B \vdash A, Y}{X, A \rightarrow B \vdash Y} \quad \frac{X, A \rightarrow B, B \vdash Y}{\rightarrow_L} \quad \frac{X \vdash Y, A \rightarrow B, B}{\rightarrow_R}\]

\[\frac{X, A \leftarrow B, A \vdash Y}{X, A \leftarrow B \vdash Y} \quad \frac{X \vdash A, A \leftarrow B, Y}{\leftarrow_L} \quad \frac{X, B \vdash A \leftarrow B, Y}{\leftarrow_R}\]
**LBInt₂ Rules**

\[
\{X\} = \{A \mid X = (A, Y) \text{ for some } A \text{ and } Y\}
\]

\[
\frac{x, A \vdash A, Y}{\text{id}}
\]

\[
\frac{X_2 \vdash Y_2, \{Y_1\}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{Y_1\} \not\subseteq \{Y_2\}
\]

\[
\frac{\{X_1\}, X_2 \vdash Y_2}{X_1 \vdash Y_1, (X_2 > Y_2)} > \{X_1\} \not\subseteq \{X_2\}
\]

\[
\frac{X, A \rightarrow B \vdash A, Y}{X, A \rightarrow B \vdash Y} \rightarrow_L
\]

\[
\frac{X \vdash Y, A \rightarrow B, B}{X \vdash Y, A \rightarrow B} \rightarrow_{R1}
\]

\[
\frac{X, A \triangleleft B, A \vdash Y}{X, A \triangleleft B \vdash Y} \triangleleft_{L1}
\]

\[
\frac{X \vdash A, A \triangleleft B, Y}{X \vdash A \triangleleft B, Y} \triangleleft_R
\]

\[
\frac{A \vdash B, \{Y\}, (X, A \triangleleft B) > Y}{X, A \triangleleft B \vdash Y} \triangleleft_{L2}
\]

\[
\frac{(X < Y, A \rightarrow B), \{X\}, A \vdash B}{X \vdash Y, A \rightarrow B} \rightarrow_{R2}
\]
Uustalu’s Example Revisited

Using LBiInt₁:

\[
\begin{array}{c}
p \vdash q, p \\
\hline
\text{Id} \\
p \vdash q, p \quad \vdash q \\
\hline
\text{Id} \\
\vdash q, p \prec q \\
\hline
(p < q), r \vdash p \prec q \\
\hline
< \\
\vdash (p < q), r \vdash r \\
\hline
\text{Id} \\
\vdash (p < q), r \vdash (p < q) \land r \\
\hline
\land_R \\
p < q \vdash r \rightarrow ((p < q) \land r) \\
\hline
\rightarrow_R \\
p \vdash q, r \rightarrow ((p < q) \land r) \\
\hline
\text{S}_L
\end{array}
\]

Using LBiInt₂:

\[
\begin{array}{c}
p \vdash q, \ldots, p \\
\hline
\text{Id} \\
p \vdash q, \ldots, p \prec q \\
\hline
\vdash (p < q, \ldots), p, r \vdash p \prec q \\
\hline
< \\
\vdash (p < q, \ldots), p, r \vdash (p < q) \land r \\
\hline
\land_R \\
p \vdash q, r \rightarrow ((p < q) \land r) \\
\hline
\rightarrow_{R2}
\end{array}
\]
Save/Restore

- \( \text{LBilInt}_1 \) vs \( \text{LBilInt}_2 \):

  \[ \begin{array}{c}
  \text{Lose context:} \\
  \frac{X, A \vdash B}{X \vdash Y, A \rightarrow B} \rightarrow_R \\
  \end{array} \]

  \[ \begin{array}{c}
  \text{Save context:} \\
  \frac{(X < Y, A \rightarrow B), \{X\}, A \vdash B}{X \vdash Y, A \rightarrow B} \rightarrow_{R^2} \\
  \end{array} \]
Save/Restore

- **LBilInt$_1$ vs LBilInt$_2$:**
  
  Lose context:
  \[
  \frac{X, A \vdash B}{X \vdash Y, A \rightarrow B} \rightarrow^R
  \]

  Save context:
  \[
  \frac{(X < Y, A \rightarrow B), \{X\}, A \vdash B}{X \vdash Y, A \rightarrow B} \rightarrow^{R_2}
  \]

- **Restore context:**
  \[
  \frac{X_2 \vdash Y_2, \{Y_1\}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{Y_1\} \not\subseteq \{Y_2\}
  \]
Save/Restore

- **LBilInt**₁ vs **LBilInt**₂:

  Lose context:
  \[
  \frac{X, A \vdash B}{X \vdash Y, A \rightarrow B} \quad \rightarrow^R
  \]

  Save context:
  \[
  \frac{(X < Y, A \rightarrow B), \{ X \}, A \vdash B}{X \vdash Y, A \rightarrow B} \quad \rightarrow^{R2}
  \]

- Restore context:
  \[
  \frac{X_2 \vdash Y_2, \{ Y_1 \}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{ Y_1 \} \not\subseteq \{ Y_2 \}
  \]

- **LBilInt**₂ completeness via translation from G**BiInt**
Save/Restore

- **LBilnt₁ vs LBilnt₂:***

  Lose context:
  \[
  \frac{X, A \vdash B}{X \vdash Y, A \rightarrow B} \quad \rightarrow R
  \]

  Save context:
  \[
  \frac{(X < Y, A \rightarrow B), \{X\}, A \vdash B}{X \vdash Y, A \rightarrow B} \quad \rightarrow R^2
  \]

- **Restore context:***

  \[
  \frac{X_2 \vdash Y_2, \{Y_1\}}{X_1, (X_2 < Y_2) \vdash Y_1} < \{Y_1\} \not\subseteq \{Y_2\}
  \]

- **LBilnt₂ completeness via translation from GBilnt***

- **GBilnt recompute rule \(\sim\) pair of LBilnt₂ save/restore rules***
Definition

A sequent \( X \vdash Y \) is saturated iff it satisfies:
Definition

A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
Saturation

Definition
A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
Saturation

Definition
A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
3. If $A \land B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{Y\}$
Saturation

Definition
A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
3. If $A \land B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{Y\}$
4. If $A \lor B \in \{X\}$ then $A \in \{X\}$ or $B \in \{X\}$
Saturation

Definition

A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
3. If $A \land B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{Y\}$
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 $B \in \{Y\}$
Definition
A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
3. If $A \land B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{Y\}$
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 $B \in \{X\}$
Definition

A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
3. If $A \land B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{Y\}$
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 $A \prec B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{X\}$
Saturation

Definition

A sequent \( X \vdash Y \) is saturated iff it satisfies:

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

Definition
A sequent $X \vdash Y$ is saturated iff it satisfies:

1. $\{X\} \cap \{Y\} = \emptyset$
2. If $A \land B \in \{X\}$ then $A \in \{X\}$ and $B \in \{X\}$
3. If $A \land B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{Y\}$
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 $A \rightarrow B \in \{Y\}$ then $A \in \{Y\}$ or $B \in \{X\}$
8. If $A \rightarrow B \in \{Y\}$ then $B \in \{Y\}$
9. If $A \rightarrow B \in \{X\}$ then $A \in \{X\}$
Definition
We classify the rules of LBiInt$_2$ into three groups:
Definition
We classify the rules of LBiInt$_2$ into three groups:

**Static Rules:** $\{id, \land_L, \land_R, \lor_L, \lor_R, \rightarrow_L, \leftarrow_R, \leftarrow_L, \rightarrow_R\}$;
**Definition**

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

- **Static Rules:** $\{id, \land_L, \land_R, \lor_L, \lor_R, \rightarrow_L, \leftarrow_R, \leftarrow_L1, \rightarrow_R1\}$;
- **Jump Rules:** $\{\leftarrow_L2, \rightarrow_R2\}$; and
Definition

We classify the rules of $\text{LBiInt}_2$ into three groups:

- **Static Rules:** $\{id, \land_L, \land_R, \lor_L, \lor_R, \rhd_L, \rhd_R, \rhd_{L1}, \rhd_{R1}\}$;

- **Jump Rules:** $\{\rhd_{L2}, \rhd_{R2}\}$; and

- **Return Rules:** $\{\lt, \gt\}$. 
Definition
We classify the rules of $\text{LBiInt}_2$ into three groups:

- Static Rules: $\{\text{id}, \land_L, \land_R, \lor_L, \lor_R, \rightarrow_L, \leftarrow_R, \leftarrow_{L1}, \rightarrow_{R1}\}$;
- Jump Rules: $\{\leftarrow_{L2}, \rightarrow_{R2}\}$; and
- Return Rules: $\{<, >\}$. 
Definition

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

**Static Rules:** $\{id, \land_L, \land_R, \lor_L, \lor_R, \rightarrow_L, \leftarrow_R, \leftarrow_{L1}, \rightarrow_{R1}\}$;

**Jump Rules:** $\{\leftarrow_{L2}, \rightarrow_{R2}\}$; and

**Return Rules:** $\{<, >\}$.

We call a sequence of static rule applications a *saturation*. 
Definition
We classify the rules of LBiInt_2 into three groups:

Static Rules: \( \{id, \land_L, \land_R, \lor_L, \lor_R, \to_L, \leftarrow R, \leftarrow L_1, \to R_1\} \);

Jump Rules: \( \{\leftarrow L_2, \to R_2\} \); and

Return Rules: \( \{<, >\} \).

We call a sequence of static rule applications a saturation.

Definition
A LBiInt_2 rule \( \rho \) is applicable to a sequent \( \gamma_0 = X_0 \vdash Y_0 \) if for every premise \( X_i \vdash Y_i \) of \( \rho \), \( \{X_i\} \not\subseteq \{X_0\} \) or \( \{Y_i\} \not\subseteq \{Y_0\} \).
**Blocking**

**Definition**
We classify the rules of LBInt\textsubscript{2} into three groups:

- **Static Rules:** $\{ id, \wedge_L, \wedge_R, \vee_L, \vee_R, \rightarrow_L, \leftarrow_R, \leftarrow_L1, \rightarrow_R1 \}$;
- **Jump Rules:** $\{ \leftarrow_L2, \rightarrow_R2 \}$; and
- **Return Rules:** $\{ <, > \}$.

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

**Definition**
A LBInt\textsubscript{2} rule $\rho$ is applicable to a sequent $\gamma_0 = X_0 \vdash Y_0$ if for every premise $X_i \vdash Y_i$ of $\rho$, $\{X_i\} \not\subseteq \{X_0\}$ or $\{Y_i\} \not\subseteq \{Y_0\}$.

**Corollary**
*Only jump and return rules are applicable to saturated sequents.*
Proof Search Strategy

**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*
Proof Search Strategy

**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
Proof Search Strategy

**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
   1. Let $\gamma_1, \ldots, \gamma_n$ be the premises of $\rho$ obtained from $\gamma_0$
**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
   1. Let $\gamma_1, \cdots, \gamma_n$ be the premises of $\rho$ obtained from $\gamma_0$
   2. Return $\land_{i=1}^{n} Prove(\gamma_i)$
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
   1. Let $\gamma_1, \cdots, \gamma_n$ be the premises of $\rho$ obtained from $\gamma_0$
   2. Return $\bigwedge_{i=1}^n \text{Prove}(\gamma_i)$
3. Else if $\text{Prove}(\gamma_1) = true$ for some premise instance $\gamma_1$ obtained from $\gamma_0$ by applying $\rho \in \{\langle L_2, \rightarrow R_2 \rangle, <, >\}$ then return true
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
   1. Let $\gamma_1, \ldots, \gamma_n$ be the premises of $\rho$ obtained from $\gamma_0$
   2. Return $\land_{i=1}^n \text{Prove}(\gamma_i)$
3. Else if $\text{Prove}(\gamma_1) = true$ for some premise instance $\gamma_1$ obtained from $\gamma_0$ by applying $\rho \in \{\langle L2, R2, <, >\}$ then return true
4. Else return false.
Linear Sequents

Definition
Linear Sequents

Definition

1. If $\Gamma/\Delta$ are sets of formulae, then $\Gamma \vdash \Delta$ is a linear sequent.
Linear Sequents

Definition

1. If $\Gamma/\Delta$ are sets of formulae, then $\Gamma \vdash \Delta$ is a linear sequent.
2. If $X \vdash Y$ is a linear sequent and $\Gamma/\Delta$ are sets of formulae, then

are linear sequents.
Linear Sequents

Definition

1. If $\Gamma/\Delta$ are sets of formulae, then $\Gamma \vdash \Delta$ is a linear sequent.
2. If $X \vdash Y$ is a linear sequent and $\Gamma/\Delta$ are sets of formulae, then
   1. $(X < Y), \Gamma \vdash \Delta$ and

   are linear sequents.
Linear Sequents

Definition

1. If $\Gamma/\Delta$ are sets of formulae, then $\Gamma \vdash \Delta$ is a linear sequent.
2. If $X \vdash Y$ is a linear sequent and $\Gamma/\Delta$ are sets of formulae, then
   1. $(X < Y), \Gamma \vdash \Delta$ and
   2. $\Gamma \vdash \Delta, (X > Y)$

are linear sequents.
Linear Sequents

**Definition**

1. If $\Gamma/\Delta$ are sets of formulae, then $\Gamma \vdash \Delta$ is a linear sequent.
2. If $X \vdash Y$ is a linear sequent and $\Gamma/\Delta$ are sets of formulae, then
   1. $(X < Y), \Gamma \vdash \Delta$ and
   2. $\Gamma \vdash \Delta, (X > Y)$

are linear sequents.

**Example**

$C \vdash B, A \rightarrow B$

$(C < B, A \rightarrow B), C, A \vdash B$

$D \vdash E, ((C < B, A \rightarrow B), C, A > B)$
Linear Sequents

Definition

1. If $\Gamma/\Delta$ are sets of formulae, then $\Gamma \vdash \Delta$ is a linear sequent.
2. If $X \vdash Y$ is a linear sequent and $\Gamma/\Delta$ are sets of formulae, then
   1. $(X < Y), \Gamma \vdash \Delta$ and
   2. $\Gamma \vdash \Delta, (X > Y)$
   are linear sequents.

Example

$C \vdash B, A \rightarrow B$
$(C < B, A \rightarrow B), C, A \vdash B$
$D \vdash E, ((C < B, A \rightarrow B), C, A > B)$

Lemma

Every $L\text{BiInt}_2$-derivation of a linear end-sequent contains only linear sequents.
Definition (Linear Sequent to List)

\[
\begin{align*}
\text{list}(\Gamma \vdash \Delta) &= \langle \Gamma, \Delta \rangle \\
\text{list}((X < Y), \Gamma \vdash \Delta) &= \text{list}(X \vdash Y) \leq \langle \Gamma, \Delta \rangle \\
\text{list}(\Gamma \vdash \Delta, (X > Y)) &= \text{list}(X \vdash Y) \geq \langle \Gamma, \Delta \rangle
\end{align*}
\]
Lists

Definition (Linear Sequent to List)

\[
\begin{align*}
\text{list}(\Gamma \vdash \Delta) &= \langle \Gamma, \Delta \rangle \\
\text{list}((X < Y), \Gamma \vdash \Delta) &= \text{list}(X \vdash Y) \leq \langle \Gamma, \Delta \rangle \\
\text{list}(\Gamma \vdash \Delta, (X > Y)) &= \text{list}(X \vdash Y) \geq \langle \Gamma, \Delta \rangle
\end{align*}
\]

Example

\[
\begin{align*}
\text{list}(C \vdash B, A \rightarrow B) &= \langle \{C\}, \{B, A \rightarrow B\} \rangle \\
\text{list}((C < B, A \rightarrow B), C, A \vdash B) &= \text{list}(C \vdash B, A \rightarrow B) \leq \langle \{C, A\}, \{B\} \rangle \\
&= \langle \{C\}, \{B, A \rightarrow B\} \rangle \leq \langle \{C, A\}, \{B\} \rangle
\end{align*}
\]
Corollary

A \textit{backward \text{LB}Int}_2 \textit{ rule application to a linear sequent } X \vdash Y \textit{ can be viewed as an operation on list}(X \vdash Y)$:
Corollary

A backward $\text{LBiInt}_2$ rule application to a linear sequent $X \vdash Y$ can be viewed as an operation on $\text{list}(X \vdash Y)$:

- Conclusion/premise is the list before/after the operation
Corollary

A backward LBInt$_2$ rule application to a linear sequent $X ⊢ Y$ can be viewed as an operation on list($X \vdash Y$):

- Conclusion/premise is the list before/after the operation
- Jump rules: append a node to the list
Corollary

A backward LBInt$_2$ rule application to a linear sequent $X \vdash Y$ can be viewed as an operation on list$(X \vdash Y)$:

- **Conclusion/premise is the list before/after the operation**
- **Jump rules**: append a node to the list
- **Static rules**: saturate the end node
Corollary

A backward LBiInt$_2$ rule application to a linear sequent $X \vdash Y$ can be viewed as an operation on list($X \vdash Y$):

- **Conclusion/premise** is the list before/after the operation
- **Jump rules**: append a node to the list
- **Static rules**: saturate the end node
- **Return rules**: remove end node, update penultimate node.
Corollary

A backward $LBiInt_2$ rule application to a linear sequent $X \vdash Y$ can be viewed as an operation on $\text{list}(X \vdash Y)$:

- Conclusion/premise is the list before/after the operation
- Jump rules: append a node to the list
- Static rules: saturate the end node
- Return rules: remove end node, update penultimate node.

Example

\[
\frac{(C < B, A \rightarrow B), C, A \vdash B}{C \vdash B, A \rightarrow B} \quad \rightarrow_{R2} \quad \frac{\langle \{C\}, \{B, A \rightarrow B\} \rangle \leq \langle \{C, A\}, \{B\} \rangle}{\langle \{C\}, \{B, A \rightarrow B\} \rangle}
\]
Termination

Lemma (Bounded Lists)

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

Let $X \vdash Y$ be any sequent encountered during proof search. Using jump rules, list($X \vdash Y$) can be extended at most $O(m^2)$ times.

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.
Lemma (Bounded Lists)

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

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.
Lemma (Bounded Lists)

Let $X \vdash Y$ be any sequent encountered during proof search. Using jump rules, list($X \vdash Y$) can be extended at most $O(m^2)$ times.

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.

Proof.
Lemma (Bounded Lists)

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

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.

Proof.

- Nodes/lists are bounded in size/length
Lemma (Bounded Lists)

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

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.

Proof.

- Nodes/lists are bounded in size/length
- Jump/return rules cannot repeatedly create/remove nodes
Termination

Lemma (Bounded Lists)

Let $X \vdash Y$ be any sequent encountered during proof search. Using jump rules, list($X \vdash Y$) can be extended at most $O(m^2)$ times.

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.

Proof.

- Nodes/lists are bounded in size/length
- Jump/return rules cannot repeatedly create/remove nodes
  - Every update adds one more (sub)formula to a node
Termination

Lemma (Bounded Lists)

Let $X \vdash Y$ be any sequent encountered during proof search. Using jump rules, list($X \vdash Y$) can be extended at most $O(m^2)$ times.

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.

Proof.

- Nodes/lists are bounded in size/length
- Jump/return rules cannot repeatedly create/remove nodes
  - Every update adds one more (sub)formula to a node
  - Eventually no subformulae can be added to any node
Termination

Lemma (Bounded Lists)

Let $X \vdash Y$ be any sequent encountered during proof search. Using jump rules, list($X \vdash Y$) can be extended at most $O(m^2)$ times.

Lemma (Saturation)

Let $X \vdash Y$ be any sequent encountered during proof search. Then the saturation process for $X \vdash Y$ terminates after $O(m)$ steps.

Theorem

The proof search strategy terminates.

Proof.

- Nodes/lists are bounded in size/length
- Jump/return rules cannot repeatedly create/remove nodes
  - Every update adds one more (sub)formula to a node
  - Eventually no subformulae can be added to any node
- Return rules are blocked
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
• BiInt presents proof-theoretic challenges
• Nested sequents allow to tame display calculi
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - LBiInt$_1$: syntactic cut-elimination
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - LBiInt\textsubscript{1}: syntactic cut-elimination
  - LBiInt\textsubscript{2}: terminating proof search
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - LBiInt\textsubscript{1}: syntactic cut-elimination
  - LBiInt\textsubscript{2}: terminating proof search
- Completeness vs cut-elimination
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - $\text{LBiInt}_1$: syntactic cut-elimination
  - $\text{LBiInt}_2$: terminating proof search
- Completeness vs cut-elimination
  - $\text{LBiInt}_1$ has elegant cut-elimination due to structural rules
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - $\text{LBilInt}_1$: syntactic cut-elimination
  - $\text{LBilInt}_2$: terminating proof search
- Completeness vs cut-elimination
  - $\text{LBilInt}_1$ has elegant cut-elimination due to structural rules
  - Current completeness of $\text{LBilInt}_2$ relies on semantics
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - LBiInt\(_1\): syntactic cut-elimination
  - LBiInt\(_2\): terminating proof search
- Completeness vs cut-elimination
  - LBiInt\(_1\) has elegant cut-elimination due to structural rules
  - Current completeness of LBiInt\(_2\) relies on semantics
  - Does LBiInt\(_2\) have direct cut-elimination?
Conclusions and Further Work

• BiInt presents proof-theoretic challenges
• Nested sequents allow to tame display calculi
  • LBiInt$_1$: syntactic cut-elimination
  • LBiInt$_2$: terminating proof search
• Completeness vs cut-elimination
  • LBiInt$_1$ has elegant cut-elimination due to structural rules
  • Current completeness of LBiInt$_2$ relies on semantics
  • Does LBiInt$_2$ have direct cut-elimination?
  • Aim to bridge the gap between LBiInt$_1$ and LBiInt$_2$
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - LBilnt₁: syntactic cut-elimination
  - LBilnt₂: terminating proof search
- Completeness vs cut-elimination
  - LBilnt₁ has elegant cut-elimination due to structural rules
  - Current completeness of LBilnt₂ relies on semantics
  - Does LBilnt₂ have direct cut-elimination?
  - Aim to bridge the gap between LBilnt₁ and LBilnt₂
- Generalising LBilnt to other logics
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - LBiInt$_1$: syntactic cut-elimination
  - LBiInt$_2$: terminating proof search
- Completeness vs cut-elimination
  - LBiInt$_1$ has elegant cut-elimination due to structural rules
  - Current completeness of LBiInt$_2$ relies on semantics
  - Does LBiInt$_2$ have direct cut-elimination?
  - Aim to bridge the gap between LBiInt$_1$ and LBiInt$_2$
- Generalising LBiInt to other logics
  - KtS4, S5 and bi-Lambek logic have similar properties to BiInt
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - \( \text{LBiInt}_1 \): syntactic cut-elimination
  - \( \text{LBiInt}_2 \): terminating proof search
- Completeness vs cut-elimination
  - \( \text{LBiInt}_1 \) has elegant cut-elimination due to structural rules
  - Current completeness of \( \text{LBiInt}_2 \) relies on semantics
  - Does \( \text{LBiInt}_2 \) have direct cut-elimination?
  - Aim to bridge the gap between \( \text{LBiInt}_1 \) and \( \text{LBiInt}_2 \)
- Generalising \( \text{LBiInt} \) to other logics
  - \( \text{KtS4, S5 and bi-Lambek logic have similar properties to BiInt} \)
  - The (almost) power of display logic with proof search?
Conclusions and Further Work

- BiInt presents proof-theoretic challenges
- Nested sequents allow to tame display calculi
  - $\text{LBiInt}_1$: syntactic cut-elimination
  - $\text{LBiInt}_2$: terminating proof search
- Completeness vs cut-elimination
  - $\text{LBiInt}_1$ has elegant cut-elimination due to structural rules
  - Current completeness of $\text{LBiInt}_2$ relies on semantics
  - Does $\text{LBiInt}_2$ have direct cut-elimination?
  - Aim to bridge the gap between $\text{LBiInt}_1$ and $\text{LBiInt}_2$
- Generalising $\text{LBiInt}$ to other logics
  - KtS4, S5 and bi-Lambek logic have similar properties to BiInt
  - The (almost) power of display logic with proof search?
  - General technique for taming display calculi?