cse371/mat371
LOGIC

Professor Anita Wasilewska

Fall 2015
LECTURE 12
Chapter 12
Gentzen Sequent Calculus \textbf{L}I for Intuitionistic Logic
Original Gentzen System \textbf{LI} for Intuitionistic Logic

Part 1

Definition of Gentzen System \textbf{LI}

The proof system \textbf{LI} for Intuitionistic Logic as presented here was published by G. Gentzen in 1935.

It was presented as a particular case of his proof system \textbf{LK} for the classical logic.

We present now the original Gentzen proof system \textbf{LI} and then we show how it can be extended to the original Gentzen system \textbf{LK}. 
Language of \textbf{LI}

Language of \textbf{LI} is

$$\mathcal{L} = \mathcal{L}_{\{\cup, \cap, \Rightarrow, \neg\}}$$

We add a new symbol $\rightarrow$ to the language and call it a Gentzen arrow.

We denote, as before, the finite sequences of formulas by Greek capital letters

$$\Gamma, \Delta, \Sigma, \ldots$$

with indices if necessary.
Language of LI

Definition Any expression

\[ \Gamma \rightarrow \Delta \]

where \( \Gamma, \Delta \in F^* \) and

\( \Delta \) consists of at most one formula

is called a LI sequent

We denote the set of all LI sequents by \( ISQ \), i.e.

\[ ISQ = \{ \Gamma \rightarrow \Delta : \Delta \text{ consists of at most one formula} \} \]
Axioms of LI

Logical Axioms of LI consist of any sequent from the set ISQ which contains a formula that appears on both sides of the sequent arrow $\rightarrow$, i.e. any sequent of the form

$$\Gamma, A, \Delta \rightarrow A$$

for $\Gamma, \Delta \in \mathcal{F}^*$
Rules of Inference of LI

The set inference rules of LI is divided into two groups: the structural rules and the logical rules.

There are three Structural Rules of LI: Weakening, Contraction, and Exchange.

**Weakening** structural rule

\[
\frac{\Gamma \rightarrow \Delta}{A, \Gamma \rightarrow \Delta}
\]

\[
\frac{\Gamma \rightarrow}{\Gamma \rightarrow A}
\]

\(A\) is called the weakening formula.

**Remember** that \(\Delta\) contains at most one formula.
Rules of Inference of LI

**Contraction** structural rule

\[
(\text{contr } \rightarrow) \quad \frac{A, A, \Gamma \rightarrow \Delta}{A, \Gamma \rightarrow \Delta}
\]

\(A\) is called the contraction formula

**Remember** that \(\Delta\) contains at most one formula

The case below is **not VALID** for LI; we list it as it will be used in the classical case

\[
(\rightarrow \text{contr}) \quad \frac{\Gamma \rightarrow \Delta, A, A}{\Gamma \rightarrow \Delta, A}
\]
Rules of Inference of LI

**Exchange** structural rule

\[
\begin{align*}
\text{exchange} \quad & \quad \Gamma_1, A, B, \Gamma_2 \rightarrow \Delta \\
\text{exchange} \quad & \quad \Gamma_1, B, A, \Gamma_2 \rightarrow \Delta
\end{align*}
\]

**Remember** that \( \Delta \) contains at most one formula

The rule below is **not VALID** for LI; we list it as it will be used in the classical case

\[
\begin{align*}
\text{exchange} \quad & \quad \Delta \rightarrow \Gamma_1, A, B, \Gamma_2 \\
\text{exchange} \quad & \quad \Delta \rightarrow \Gamma_1, B, A, \Gamma_2
\end{align*}
\]
Rules of Inference of LI

Logical Rules

Conjunction rules

$$(\cap \rightarrow) \quad \frac{A, B, \Gamma \rightarrow \Delta}{(A \cap B), \Gamma \rightarrow \Delta},$$

$$(\rightarrow \cap) \quad \frac{\Gamma \rightarrow A ; \Gamma \rightarrow B}{\Gamma \rightarrow (A \cap B)}$$

Remember that $\Delta$ contains at most one formula
Rules of Inference of LI

Disjunction rules

\[(\to \cup)_1\]

\[
\begin{array}{c}
\Gamma \to A \\
\hline
\Gamma \to (A \cup B)
\end{array}
\]

\[(\to \cup)_2\]

\[
\begin{array}{c}
\Gamma \to B \\
\hline
\Gamma \to (A \cup B)
\end{array}
\]

\[(\cup \to)\]

\[
\begin{array}{c}
A, \Gamma \to \Delta \quad ; \quad B, \Gamma \to \Delta \\
\hline
(A \cup B), \Gamma \to \Delta
\end{array}
\]

Remember that \(\Delta\) contains at most one formula
Rules of Inference of \textbf{LI}

Implication rules

\[
\begin{align*}
(\rightarrow\Rightarrow) & \quad A, \Gamma \rightarrow B \\
& \quad \Gamma \rightarrow (A \Rightarrow B)
\end{align*}
\]

\[
\begin{align*}
(\Rightarrow\rightarrow) & \quad \Gamma \rightarrow A \quad ; \quad B, \Gamma \rightarrow \Delta \\
& \quad (A \Rightarrow B), \Gamma \rightarrow \Delta
\end{align*}
\]

Remember that $\Delta$ contains at most one formula
Gentzen System \textbf{LI}

Negation rules

\[
(\neg \rightarrow) \quad \frac{\Gamma \rightarrow A}{\neg A, \Gamma \rightarrow} \\
\]

\[
(\rightarrow \neg) \quad \frac{A, \Gamma \rightarrow}{\Gamma \rightarrow \neg A} \\
\]

We define the Gentzen System \textbf{LI} as

\[
\text{LI} = (\mathcal{L}, \ ISQ, \ LA, \ Structural\ rules, \ Logical\ rules) \\
\]
LK - Original Gentzen system for Classical Propositional Logic
Classical Gentzen System \textbf{LK}

\textbf{Language of LK}

\[ \mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}} \quad \text{and} \quad \mathcal{E} = SQ \]

\text{for}

\[ SQ = \{ \Gamma \rightarrow \Delta : \Gamma, \Delta \in \mathcal{F}^* \} \]

\textbf{Axioms of LK} \quad \text{any sequent of the form}

\[ \Gamma_1, A, \Gamma_2 \rightarrow \Gamma_3, A, \Gamma_4 \]
Classical Gentzen System $\textbf{LK}$

Rules of inference of LK

1. We adopt all rules of $\textbf{LI}$ with no restriction that the sequence $\Delta$ in the succedent of the sequence is at most one formula.

2. We add the following structural rules to the system $\textbf{LI}$.

   **Contraction rule**

   \[
   (\rightarrow \text{contr}) \quad \frac{\Gamma \rightarrow \Delta, A, A}{\Gamma \rightarrow \Delta, A}
   \]

2. We add one more

   **Exchange rule**

   \[
   (\rightarrow \text{exch}) \quad \frac{\Delta \rightarrow \Gamma_1, A, B, \Gamma_2}{\Delta \rightarrow \Gamma_1, B, A, \Gamma_2}
   \]
Observe that the added rules become obsolete in LI.

The rules of inference of LK are hence as follows:

**Weakening (Structural Rule)**

\[
\begin{align*}
\text{(weak →)} & \quad \frac{\Gamma \rightarrow \Delta}{A, \Gamma \rightarrow \Delta} \\
\text{→ weak} & \quad \frac{\Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, A}
\end{align*}
\]

**Contraction (Structural Rule)**

\[
\begin{align*}
\text{(contr →)} & \quad \frac{A, A, \Gamma \rightarrow \Delta}{A, \Gamma \rightarrow \Delta} \\
\text{→ contr} & \quad \frac{\Gamma \rightarrow \Delta, A, A}{\Gamma \rightarrow \Delta, A}
\end{align*}
\]
Classical Gentzen System $\textbf{LK}$

**Exchange** Structural Rule

\[
\begin{align*}
\text{(exch} \rightarrow & \text{)} & \Gamma_1, A, B, \Gamma_2 & \rightarrow & \Delta \\
& \Gamma_1, B, A, \Gamma_2 & \rightarrow & \Delta \\
\text{(
rightarrow \text{exch}) } & \Delta & \rightarrow & \Gamma_1, A, B, \Gamma_2 \\
& \Delta & \rightarrow & \Gamma_1, B, A, \Gamma_2
\end{align*}
\]
Classical Gentzen System \( \textbf{LK} \)

Logical Rules

**Conjunction rules**

\[
\begin{align*}
(\cap \rightarrow) & \quad A, B, \Gamma \rightarrow \Delta \\
& \quad (A \cap B), \Gamma \rightarrow \Delta \\
(\rightarrow \cap) & \quad \Gamma \rightarrow \Delta, A \\
& \quad \Gamma \rightarrow \Delta, B, \Delta \\
& \quad \Gamma \rightarrow \Delta, (A \cap B)
\end{align*}
\]

**Disjunction rules**

\[
\begin{align*}
(\rightarrow \cup) & \quad \Gamma \rightarrow \Delta, A, B \\
& \quad \Gamma \rightarrow \Delta, (A \cup B)
\end{align*}
\]

\[
\begin{align*}
(\cup \rightarrow) & \quad A, \Gamma \rightarrow \Delta \\
& \quad B, \Gamma \rightarrow \Delta \\
& \quad (A \cup B), \Gamma \rightarrow \Delta
\end{align*}
\]
Classical Gentzen System \textbf{LK}

Implication rules

\[
\begin{align*}
\text{(\(\rightarrow\rightarrow\))} \quad & \quad A, \Gamma \rightarrow \Delta, B \\
& \Rightarrow \Gamma \rightarrow \Delta, (A \Rightarrow B) \\
\text{(\(\Rightarrow\rightarrow\))} \quad & \quad \Gamma \rightarrow \Delta, A; \quad B, \Gamma \rightarrow \Delta \\
& \Rightarrow (A \Rightarrow B), \Gamma \rightarrow \Delta
\end{align*}
\]
Classical Gentzen System $\textbf{LK}$

Negation rules

$\neg \rightarrow$ 

$\frac{\Gamma \rightarrow \Delta, A}{\neg A, \Gamma \rightarrow \Delta}$

$\rightarrow \neg$

$\frac{A, \Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, \neg A}$

We define formally

$\textbf{LK} = (\mathcal{L}, SQ, LA, \text{ Structural rules, Logical rules})$
Gentzen Sequent Calculus $\mathbf{L I}$ for Intuitionistic Logic
Part 2
Decomposition Trees in LI

Search for proofs in LI is a much more complicated process than the one in classical logic systems RS or GL. In all systems the proof search procedure consists of building the decomposition trees.

Remark 1
In RS the decomposition tree $T_A$ of any formula $A$ is always unique.

Remark 2
In GL the "blind search" defines, for any formula $A$ a finite number of decomposition trees,

Nevertheless, it can be proved that the search can be reduced to examining only one of them, due to the absence of structural rules.
Decomposition Trees in LI

Remark 3
In LI the structural rules play a vital role in the proof construction and hence, in the proof search.

The fact that a given decomposition tree ends with an non-axiom leaf does not always imply that does not exist.

It might only imply that our search strategy was not good.

The problem of deciding whether a given formula A does, or does not have a proof in LI becomes more complex then in the case of Gentzen system for classical logic.
Examples

Example 1

Determine whether

$$\vdash_{LI} ((\neg A \cap \neg B) \Rightarrow \neg(A \cup B))$$

Observe that

If we find a decomposition tree of $A$ in $LI$ such that all its leaves are axiom, we have a proof, i.e.

$$\vdash_{LI} A$$

If all possible decomposition trees have a non-axiom leaf then the proof of $A$ in $LI$ does not exist, i.e.

$$\not\vdash_{LI} A$$
Examples

Consider the following decomposition tree  $T_{1A}$

\[\rightarrow ((\neg A \land \neg B) \Rightarrow (\neg (A \cup B)))\]

$\lor (\Rightarrow)$

$(\neg A \land \neg B) \rightarrow \neg (A \cup B)$

$\lor (\neg)$

$(\neg A \land \neg B), (A \cup B) \rightarrow$

$\lor (\lor)$

$\neg A, \neg B, (A \cup B) \rightarrow$

$\lor (\neg)$

$\neg B, (A \cup B) \rightarrow A$

$\lor (\lor)$

$\neg B, (A \cup B) \rightarrow$

$\lor (\neg)$

$(A \cup B) \rightarrow B$

$\lor (\lor)$

$A \rightarrow B$

$B \rightarrow B$

non – axiom

axiom
Examples

The tree $T_{1_A}$ has a non-axiom leaf, so it does not constitute a proof in LI.

Observe that the decomposition tree in LI is not always unique.

Hence this fact does not yet prove that a proof of $A$ does not exist.

Consider the following decomposition tree $T_{2_A}$.
\[\rightarrow ((\neg A \cap \neg B) \Rightarrow (\neg (A \cup B)))\]

\[\mid (\rightarrow \Rightarrow)\]

\[\neg A \cap \neg B \rightarrow \neg (A \cup B)\]

\[\mid (\rightarrow \neg)\]

\[(A \cup B), (\neg A \cap \neg B) \rightarrow\]

\[\mid (exch \rightarrow)\]

\[\neg A \cap \neg B, (A \cup B) \rightarrow\]

\[\mid (\cap \rightarrow)\]

\[\neg A, \neg B, (A \cup B) \rightarrow\]

\[\mid (exch \rightarrow)\]

\[\neg A, (A \cup B), \neg B \rightarrow\]

\[\mid (exch \rightarrow)\]

\[(A \cup B), \neg A, \neg B \rightarrow\]

\[\bigwedge (\cup \rightarrow)\]

\[A, \neg A, \neg B \rightarrow\]

\[\mid (exch \rightarrow)\]

\[\neg A, A, \neg B \rightarrow\]

\[\mid (\neg \rightarrow)\]

\[\neg A, \neg B \rightarrow A\]

\[axiom\]

\[B, \neg A, \neg B \rightarrow\]

\[\mid (exch \rightarrow)\]

\[B, \neg B, \neg A \rightarrow\]

\[\mid (exch \rightarrow)\]

\[\neg B, B, \neg A \rightarrow\]

\[\mid (\neg \rightarrow)\]

\[B, \neg A \rightarrow B ; axiom\]
Examples

All leaves of $T_{2A}$ are axioms and hence $T_{2A}$ is a proof in $L_l$

Hence we proved that

$$\vdash_{L_l} \left( (\neg A \cap \neg B) \Rightarrow \neg (A \cup B) \right)$$
Examples

Example 2: Show that

1. $\vdash_{LI} (A \Rightarrow \neg \neg A)$

2. $\not\vdash_{LI} (\neg \neg A \Rightarrow A)$

Solution of 1.
We construct some, or all decomposition trees of

$\rightarrow (A \Rightarrow \neg \neg A)$

The tree $T_A$ that ends with all axioms leaves is a proof of $A$ in $LI$
Examples

We construct \( T_A \) as follows

\[
\rightarrow (A \Rightarrow \neg \neg A)
\]

\[
\mid (\rightarrow \Rightarrow)
\]

\[
A \rightarrow \neg \neg A
\]

\[
\mid (\rightarrow \neg)
\]

\[
\neg A, A \rightarrow
\]

\[
\mid (\neg \rightarrow)
\]

\[
A \rightarrow A
\]

axiom

All leaves of \( T_A \) are axioms what proves that we have found a proof

We don’t need to construct any other decomposition trees.
Examples

**Solution** of 2.

In order to prove that

\[ \not \not A \Rightarrow A \]

we have to construct all decomposition trees of

\[ \rightarrow (\not \not A \Rightarrow A) \]

and show that each of them has an non-axiom leaf.
Examples

Here is $T^1_A$

\[ \rightarrow (\neg\neg A \Rightarrow A) \]

\[ | (\rightarrow \Rightarrow) \]

_one of 2 choices_

\[ \neg\neg A \rightarrow A \]

\[ | (\rightarrow \text{weak}) \]

_one of 3 choices_

\[ \neg\neg A \rightarrow \]

\[ | (\neg \rightarrow) \]

_one of 3 choices_

\[ \rightarrow \neg A \]

\[ | (\rightarrow \neg) \]

_one of 2 choices_

\[ A \rightarrow \]

_non–axiom_
Here is $T_{2A}$

$$\rightarrow (\neg\neg A \Rightarrow A)$$

$| (\rightarrow \Rightarrow)$ \ \textit{one of 2 choices}$

$\neg\neg A \rightarrow A$

$| (\text{contr} \rightarrow)$ \ \textit{second of 2 choices}$

$\neg\neg A, \neg\neg A \rightarrow A$

$| (\rightarrow \text{weak})$ \ \textit{first of 2 choices}$

$\neg\neg A, \neg\neg A \rightarrow$

$| (\rightarrow)$ \ \textit{first of 2 choices}$

$\neg\neg A \rightarrow \neg A$

$| (\rightarrow \neg)$ \ \textit{one of 2 choices}$

$A, \neg\neg A \rightarrow$

$| (\text{exch} \rightarrow)$ \ \textit{one of 2 choices}$

$\neg\neg A, A \rightarrow$

$| (\neg \rightarrow)$ \ \textit{one of 2 choices}$

$A \rightarrow \neg A$

$| (\rightarrow \neg)$ \ \textit{first of 2 choices}$

$A, A \rightarrow$

\text{non – axiom}
Structural Rules

We can see from the above decomposition trees that the "blind" construction of all possible trees only leads to more complicated trees.

This is due to the presence of structural rules "blind" application of the rule (contr →) gives always an infinite number of decomposition trees.

In order to decide that none of them will produce a proof we need some extra knowledge about patterns of their construction, or just simply about the number of useful of application of structural rules within the proofs.
Structural Rules

In this case we can just make an "external" observation that the our first tree $T_{1_A}$ is in a sense a minimal one. It means that all other trees would only complicate this one in an inessential way, i.e. the we will never produce a tree with all axioms leaves.

One can formulate a deterministic procedure giving a finite number of trees, but the proof of its correctness is needed and that requires some extra knowledge.

Within the scope of this book we accept the "external explanation as a sufficient solution, provided its correctness had been proved elsewhere.
Structural Rules

As we can see from the above examples the structural rules and especially the \((\text{contr} \rightarrow)\) rule complicates the proof searching task.

Both Gentzen type proof systems RS and GL from the previous chapter don’t contain the structural rules. They also are as we have proved, complete with respect to classical semantics.

The original Gentzen system LK which does contain the structural rules is also, as proved by Gentzen, complete.
Structural Rules

Hence all three classical proof systems RS, GL, LK are equivalent.

This proves that the structural rules can be eliminated from the system LK.

A natural question of elimination of structural rules from the Intuitionistic Gentzen system LI arises.

The following example illustrates the negative answer.
Connection Between Classical and Intuitionistic Logics

Here is the connection between Intuitionistic logic and the Classical one

Theorem 1
For any formula \( A \in \mathcal{F} \),

\[ \models A \quad \text{if and only if} \quad \vdash_{I} \neg \neg A \]

where
\[ \models A \] means that \( A \) is a classical tautology
\[ \vdash_{IS} A \] means that \( A \) is Intuitionistically provable in any Intuitionistically complete proof system \( IS \)
A Gentzen system $\textbf{LI}$ has been proved to be 
**Intuitionistically complete** so have that the following

**Theorem 2**
For any formula $A \in \mathcal{F}$,
\[ \models A \text{ if and only if } \vdash_{\textbf{LI}} \neg
\neg A \]
Example

Example 3
Obviously

\[ \vdash (\neg\neg A \Rightarrow A) \]

so by Theorem 2 we must have that

\[ \vdash_{LI} \neg(\neg\neg A \Rightarrow A) \]

We are going to prove now that the structural rule \((\text{contr} \rightarrow)\) is essential to the existence of the proof, i.e.

We show now that the formula \(\neg(\neg\neg A \Rightarrow A)\) is not provable in LI without the rule \((\text{contr} \rightarrow)\).

The following decomposition tree \(T_A\) is a proof of \(A = \neg(\neg\neg A \Rightarrow A)\) in LI with use of the contraction rule \((\text{contr} \rightarrow)\).
\[ \rightarrow \neg(\neg A \Rightarrow A) \]
\[ | (\rightarrow \neg) \]
\[ \neg(\neg A \Rightarrow A) \rightarrow \]
\[ | (\text{contr} \rightarrow) \]
\[ \neg(\neg A \Rightarrow A), \neg(\neg A \Rightarrow A) \rightarrow \]
\[ | (\neg \rightarrow) \]
\[ \neg(\neg A \Rightarrow A) \rightarrow (\neg A \Rightarrow A) \]
\[ | (\rightarrow \Rightarrow) \]
\[ \neg(\neg A \Rightarrow A), \neg A \rightarrow A \]
\[ | (\rightarrow \text{weak}) \]
\[ \neg(\neg A \Rightarrow A), \neg A \rightarrow \]
\[ | (\text{exch} \rightarrow) \]
\[ \neg A, \neg(\neg A \Rightarrow A) \rightarrow \]
\[ | (\neg \rightarrow) \]
\[ \neg(\neg A \Rightarrow A) \rightarrow \neg A \]
\[ | (\rightarrow \neg) \]
\[ A, \neg(\neg A \Rightarrow A) \rightarrow \]
\[ | (\text{exch} \rightarrow) \]
\[ \neg(\neg A \Rightarrow A), A \rightarrow \]
\[ | (\neg \rightarrow) \]
\[ A \rightarrow (\neg A \Rightarrow A) \]
\[ | (\rightarrow \Rightarrow) \]
\[ \neg A, A \rightarrow A \]

\textit{axiom}
Contraction Rule

Assume now that the Contraction rule ($\text{contr} \rightarrow$) is not available.

All possible decomposition trees are as follows:

Tree $T_1 A$

\[
\begin{align*}
\rightarrow & \equiv A \\
| & \equiv \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv \neg \neg A \Rightarrow \neg \neg A \\
| & \equiv \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

\[
\begin{align*}
\rightarrow & \equiv (\neg \neg A \Rightarrow \neg A) \\
| & \equiv \neg \neg \neg
\end{align*}
\]

non - axiom
Contraction Rule

The next is \( T_{2A} \)

\[ \rightarrow \neg \neg (\neg \neg A \Rightarrow A) \]

\[ | (\rightarrow \neg) \]

\[ \neg (\neg \neg A \Rightarrow A) \rightarrow \]

\[ | (\neg \rightarrow) \]

\[ | (\neg \rightarrow) \]

\[ \rightarrow (\neg \neg A \Rightarrow A) \]

\[ | (\rightarrow \text{weak}) \]

\[ \rightarrow \]

\textit{non – axiom}
The next is $\mathbf{T}3_A$

$\rightarrow \neg \neg (\neg \neg A \Rightarrow A) \quad | \quad \rightarrow \text{weak} \quad | \quad \rightarrow$

non-axiom
Contraction Rule

The last one is $T4_A$

\[\rightarrow \neg\neg\neg\neg A \Rightarrow A\]
\[\mid (\rightarrow \neg)\]
\[\neg\neg\neg\neg A \Rightarrow A \rightarrow\]
\[\mid (\neg \rightarrow)\]
\[\rightarrow (\neg\neg A \Rightarrow A)\]
\[\mid (\rightarrow \Rightarrow)\]

\[\neg\neg A \rightarrow A\]
\[\mid (\rightarrow weak)\]
\[\neg\neg A \rightarrow\]
\[\mid (\neg \rightarrow)\]
\[\rightarrow \neg A\]
\[\mid (\rightarrow weak)\]

\[\rightarrow\]

non–axiom
Contraction Rule

We have considered all possible decomposition trees that do not involve the Contraction Rule and none of them was a proof.

This shows that the formula

\[ \neg\neg\neg\neg A \Rightarrow A \]

is not provable in LI without \((contr \rightarrow)\) rule, i.e. that

Fact

The Contraction Rule can’t be eliminated from LI.
Exercise

Use Gentzen system $\mathbf{LI}$ to prove the following

**Theorem (Gödel, Gentzen)**

A disjunction $(A \cup B)$ is **intuitionistically provable** if and only if either $A$ or $B$ is **intuitionistically provable** i.e.

$$\vdash_I (A \cup B) \text{ if and only if } \vdash_I A \text{ or } \vdash_I B$$
Proof Search Heuristic Method

Before we define a heuristic method of searching for proof in LI let’s put together some observations.

Observation 1: the logical rules of LI are similar to those in Gentzen type classical formalizations we examined in previous chapters in a sense that each of them introduces a logical connective.

Observation 2: The process of searching for a proof is, as before a decomposition process in which we use the inverse of logical and structural rules as decomposition rules.

Observation 3: We write our proofs in as trees, instead of sequences of expressions, so the proof search process is a process of building a decomposition tree.

To facilitate the process we write, as before, the decomposition rules, structural rules in a ”tree ” form.
Proof Search Heuristic Method

We define, as before the notion of decomposable and indecomposable formulas and sequents as follows

**Decomposable formula** is any formula of the degree $\geq 1$

**Decomposable sequent** is any sequent that contains a decomposable formula

**Indecomposable formula** is any formula of the degree 0, i.e. any propositional variable

**Indecomposable sequent** is a sequent formed from indecomposable formulas only.
Proof Search Heuristic Method

**Decomposition tree** $T_A$ construction for a given a formula $A \in calF$ is as follows:

**Root** of the tree is the sequent $\rightarrow A$.

Given a node $n$ of the tree we *identify* a decomposition rule applicable at this node and write its *premisses* as the *leaves* of the node $n$.

*We stop* the decomposition process when we obtain *axioms* on all branches or all leaves of the tree are *indecomposable*. 
Proof Search Heuristic Method

Observation 4
We can see from previous examples of decomposition trees that the above “blind” construction of all possible trees only leads to more complicated trees, due to the presence of structural rules.

Observation 5
The “blind” application of structural rule \((\text{contr} \rightarrow)\) gives an infinite number of infinite decomposition trees. In order to decide that none of them would produce a proof we need some extra knowledge about patterns of their construction, or just simply about the number useful of application of structural rules within the search for the proofs.
Proof Search Heuristic Method

One can formulate a **deterministic procedure** (and we will do so) giving a **finite number of trees**

But the **proof of correctness** of such procedure requires some **extra knowledge** and theorems to be proved

We are going to discuss here a **motivation** and argue validity of such a **heuristic**

The main point is, as we can see from our examples, that the structural rules and especially the \( (\text{contr} \rightarrow) \) rule complicate in often useless way the proof searching task
Proof Search Heuristic Method

Observation 6
Our goal while constructing the decomposition tree is to obtain axiom or indecomposable leaves.

With respect to this goal the use logical decomposition rules has a priority over the use of the structural rules.

We use this information while describing the proof search heuristic.
Proof Search Heuristic Method

Observation 7
All logical decomposition rules ($\circ \rightarrow$), where $\circ$ denotes any connective, must have a formula we want to decompose as the first formula at the decomposition node.

It means that if we want to decompose a formula $\circ A$ the node must have a form $\circ A, \Gamma \rightarrow \Delta$

Remember: order of decomposition is important.
Also sometimes it is necessary to decompose a formula within the sequence $\Gamma$ first, before decomposing $\circ A$ in order to find a proof.
Proof Search Heuristic Method

For example, consider two nodes

\[ n_1 = \neg\neg A, (A \cap B) \rightarrow B \]

and

\[ n_2 = (A \cap B), \neg\neg A \rightarrow B \]

We are going to see that the results of decomposing \( n_1 \) and \( n_2 \) differ dramatically

Let's decompose the node \( n_1 \)

Observe that the only way to be able to decompose the formula \( \neg\neg A \) is to use the rule (\( \rightarrow \) weak) as a first step.

The two possible decomposition trees that starts at the node \( n_1 \) are as follows
Proof Search Heuristic Method

First Tree

$T_{1n_1}$

$\neg\neg A, (A \cap B) \rightarrow B$

$\mid (\rightarrow weak)$

$\neg\neg A, (A \cap B) \rightarrow$

$\mid (\neg \rightarrow)$

$(A \cap B) \rightarrow \neg A$

$\mid (\cap \rightarrow)$

$A, B \rightarrow \neg A$

$\mid (\rightarrow \neg)$

$A, A, B \rightarrow$

*non – axiom*
Proof Search Heuristic Method

Second Tree

$T_{2_{n_1}}$

$\neg\neg A, (A \cap B) \rightarrow B$

$| (\rightarrow \text{weak})$

$\neg\neg A, (A \cap B) \rightarrow$

$| (\neg \rightarrow)$

$(A \cap B) \rightarrow \neg A$

$| (\rightarrow \neg)$

$A, (A \cap B) \rightarrow$

$| (\cap \rightarrow)$

$A, A, B \rightarrow$

non – axiom
Let's now decompose the node \( n_2 \)
Observe that following our Observation 6 we start by decomposing the formula \((A \cap B)\) by the use of the rule \((\cap \rightarrow)\) as the first step
A decomposition tree that starts at the node \( n_2 \) is as follows

\[
T_{n_2}
\]

\[
(A \cap B), \neg \neg A \rightarrow B
\]

\[
| (\cap \rightarrow)
\]

\[
A, B, \neg \neg A \rightarrow B
\]

axiom

This proves that the node \( n_2 \) is provable in \( \text{LI} \), i.e.

\[
\vdash_{\text{LI}} (A \cap B), \neg \neg A \rightarrow B
\]
Proof Search Heuristic Method

Observation 8
The use of structural rules is important and necessary while we search for proofs.
Nevertheless we have to use them on the "must" basis and set up some guidelines and priorities for their use.

For example, the use of weakening rule discharges the weakening formula, and hence we might lose an information that may be essential to finding the proof.

We should use the weakening rule only when it is absolutely necessary for the next decomposition steps.
Proof Search Heuristic Method

Hence, the use of weakening rule ($\rightarrow$ weak) can, and should be restricted to the cases when it leads to possibility of the future use of the negation rule ($\neg \rightarrow$)

This was the case of the decomposition tree $T_{1n_1}$
We used the rule ($\rightarrow$ weak) as an necessary step, but it discharged too much information and we didn’t get a proof, when proof on this node existed
Proof Search Heuristic Method

Here is such a proof

\[ T_{3n_1} \]

\[ \neg\neg A, (A \cap B) \rightarrow B \]
\[ \mid (exch \rightarrow) \]
\[ (A \cap B), \neg\neg A \rightarrow B \]
\[ \mid (\cap \rightarrow) \]
\[ A, B, \neg\neg A \rightarrow B \]
\[ \text{axiom} \]
Proof Search Heuristic Method

Method
For any $A \in \mathcal{F}$ we construct the set of decomposition trees $T_{\rightarrow A}$ following the rules below.

1. Use first logical rules where applicable.
2. Use $(\text{exch} \rightarrow)$ rule to decompose, via logical rules, as many formulas on the left side of $\rightarrow$ as possible.
   
   **Remember** that the order of decomposition matters! so you have to cover different choices.
3. Use $(\rightarrow \text{ weak})$ only on a "must" basis and in connection with $(\neg \rightarrow)$ rule.
4. Use $(\text{contr} \rightarrow)$ rule as the last recourse and only to formulas that contain $\neg$ as a main connective.
5. Let’s call a formula $A$ to which we apply $(\text{contr} \rightarrow)$ rule a contraction formula we need to consider are the formulas containing $\neg$ between theirs logical connectives.
Proof Search Heuristic Method

7. Within the process of construction of all possible trees use \((\text{contr} \rightarrow)\) rule only to contraction formulas.

8. Let \(C\) be a contraction formula appearing on a node \(n\) of the decomposition tree of \(T \rightarrow A\).
   For any contraction formula \(C\), any node \(n\), we apply \((\text{contr} \rightarrow)\) rule the the formula \(C\) at most as many times as the number of sub-formulas of \(C\).
   If we find a tree with all axiom leaves we have a proof, i.e. \(\vdash_{LI} A\).
   If all trees (finite number) have a non-axiom leaf we have proved that proof of \(A\) does not exist, i.e. \(\nabla_{LI} A\).