

# Gentzen Sequent Calculus for Intuitionistic Logic

## PART 3: Proof Search Heuristics

**Before we define** a heuristic method of searching for proof in **LI** let's make 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 hence a decomposition process in which we use the inverse of logical and structural rules as decomposition rules.

**For example** the implication rule:

$$(\rightarrow\Rightarrow) \frac{A, \Gamma \longrightarrow B}{\Gamma \longrightarrow (A \Rightarrow B)}$$

becomes an implication decomposition rule  
(we use the same name  $(\rightarrow\Rightarrow)$  in both cases)

$$(\rightarrow\Rightarrow) \frac{\Gamma \longrightarrow (A \Rightarrow B)}{A, \Gamma \longrightarrow B}.$$

**Observation 3** : we write our proofs in a form of 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 the decomposition rules in a "tree" form. For example the the above implication decomposition rule is written as follows

$$\Gamma \longrightarrow (A \Rightarrow B)$$

$$| (\rightarrow \Rightarrow)$$

$$A, \Gamma \longrightarrow B.$$

**The two premisses** implication rule  $(\Rightarrow \rightarrow)$  written as the tree decomposition rule becomes

$$(A \Rightarrow B), \Gamma \longrightarrow$$

$$\wedge (\Rightarrow \rightarrow)$$

$$\Gamma \longrightarrow A \quad B, \Gamma \longrightarrow$$

**Observation 4** : we stop the decomposition process when we obtain an axiom or indecomposable leaf. The indecomposable leaf is a sequent built from indecomposable formulas only, i.e. formulas that do not contain logical connectives.

**Observation 5** : 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 and we use this information while describing the proof search heuristic.

**Observation 6** : 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.

**When we decompose** a formula  $\circ A$ , the node must have a form  $\circ A, \Gamma \longrightarrow \Delta$ . Sometimes it is necessary to decompose a formula within the sequence  $\Gamma$  first in order to find a proof.

**For example,** consider two nodes

$$n_1 = \neg\neg A, (A \cap B) \longrightarrow B$$

and

$$n_2 = (A \cap B), \neg\neg A \longrightarrow B.$$

**We are** going to see that the results of decomposing  $n_1$  and  $n_2$  differ dramatically.

**We 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*) first.

**The two possible** decomposition trees that starts at the node  $n_1$  are as follows.

**T1** $_{n_1}$

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

| ( $\rightarrow$  *weak*)

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

| ( $\neg \rightarrow$ )

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

| ( $\cap \rightarrow$ )

$A, B \longrightarrow \neg A$

| ( $\rightarrow \neg$ )

$A, A, B \longrightarrow$

*non - axiom*

Next tree is

**T2**<sub>n<sub>1</sub></sub>

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

| ( $\rightarrow$  weak)

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

| ( $\neg \rightarrow$ )

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

| ( $\rightarrow \neg$ )

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

| ( $\cap \rightarrow$ )

$A, A, B \longrightarrow$

*non - axiom*

**Now we decompose** the node  $n_2$ .

**Observe** that following Observation 5 we start by decomposing the formula  $(A \cap B)$  by the use of the rule  $(\cap \rightarrow)$  first.

**A decomposition tree** that starts at the node  $n_2$  is as follows.

$$\mathbf{T}_{n_2}$$
$$(A \cap B), \neg\neg A \longrightarrow B$$
$$| (\cap \rightarrow)$$
$$A, B, \neg\neg A \longrightarrow B$$

*axiom*

**This proves** that the node  $n_2$  is provable in **LI**, i.e.

$$\vdash_{\mathbf{LI}} (A \cap B), \neg\neg A \longrightarrow B.$$

**Of course,** we have also that the node  $n_1$  is also provable in **LI**, as one can obtain the node  $n_2$  from it by the use of the rule (*exch*  $\rightarrow$ ).

**Observation 7:** the use of structural rules are 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,** use of weakening rule discharges the weakening formula, and hence an information that may be essential to the proof.

**We should use** it only when it is absolutely necessary for the next decomposition steps.

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

**In the case of** the decomposition tree  $\mathbf{T1}_{n_1}$  we used it as an necessary step, but still it discharged too much information and we didn't get a proof, when proof of the node existed.

**In fact,** the first rule in our search should have been the exchange rule, followed by the conjunction rule (no information discharge!) not the weakening (discharge of information) followed by negation rule.

**The full proof** of the node  $n_1$  is the following.

**T3** <sub>$n_1$</sub>

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

| (*exch*  $\longrightarrow$ )

**T2** <sub>$A$</sub>

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

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

| ( $\cap \longrightarrow$ )

$A, B, \neg\neg A \longrightarrow B$

*axiom*

**As a result** of the observations 1- 7 we adopt the following heuristics for proof search in **LI**.

## Decomposition Tree Generation rules.

1. Use first rules logical rules where applicable without the use of ( $\rightarrow$  *weak*).
2. Use (*exch*  $\rightarrow$ ) rule to decompose as many formulas on the left side of  $\rightarrow$  as possible.
3. Use ( $\rightarrow$  *weak*) only on a "must" basis in connection with ( $\neg \rightarrow$ ).
4. Use (*contr*  $\rightarrow$ ) rule as the last recourse and only to formulas that contain  $\neg$  or  $\Rightarrow$  as connectives.
5. Within the process use (*contr*  $\rightarrow$ ) rule only a finite number of times, no more times that number of all sub-formulas of the formula you are building the tree for.