6
Experimenting with mimp-graphs in other logics.

In this chapter, we start with a brief overview of Deep Inference, focusing on the propositional fragment only. Then, we present our proof-graphs definition for SKS_g, a deductive system for classical propositional logic, that is presented in the calculus of structures (Brünnler 2004). Thereafter we will move to describe the Bi-intuitionist Logic and present a proof-graph representation for this logic.

6.1
Proof-graph for Deep Inference

Deep inference is a proof-theoretic methodology where proofs can be freely composed by the logical operators, that is inference rules are applied anywhere deep inside a formula, not only at the main connective, contrarily to traditional proof systems, such as natural deduction and the sequent calculus (Gentzen 1969).

In this section, we overview a formalism which allows deep inference based on a deductive system for classical propositional logic called SKS_g, that is presented in the calculus of structures (Brünnler 2004). The translation of derivations of a Gentzen-Schütte sequent system into this system, and vice versa, establishes soundness and completeness with respect to classical propositional logic as well as cut elimination.

6.1.1
The system SKS_g

As presented in (Brünnler 2004), SKS_g is defined below.

Formulas for propositional logic are generated by the grammar

$$S ::= f \mid t \mid a \mid [\,\underbrace{S, \ldots, S}_{>0}\,] \mid (\,\underbrace{S, \ldots, S}_{>0}\,) \mid \bar{S},$$

where f and t are the units *false* and *true*, $[S, \ldots, S]$ is a *disjuntion* and $(S, \ldots, S)$ is a *conjunction.* Atoms are denoted by $a, b, \ldots$. Formulas are denoted by S, P, Q, R, T, U, V and W, and $\bar{S}$ is the negation of the formula S. Formula contexts, denoted by $S\{\ \}$, are formulas with one occurrence of $\{\ \}$, the empty

Associativity:	$[\vec{R},[\vec{T}],\vec{U}] = [\vec{R},\vec{T},\vec{U}]$ $\quad (\vec{R},(\vec{T}),\vec{U}) = (\vec{R},\vec{T},\vec{U})$
Commutativity:	$[R,T] = [T,R]$ $\quad (R,T) = (T,R)$
Units:	$[t,t] = t$ $\quad (f,f) = f$ $\quad (t,R) = R$ $\quad [f,R] = R$
Negation:	$\bar{f} = t$ $\quad \bar{t} = f$ $\quad \overline{[R,T]} = (\bar{R},\bar{T})$ $\overline{(R,T)} = [\bar{R},\bar{T}]$ $\quad \bar{\bar{R}} = R$
Context Closure:	if $R = T$ then $S\{R\} = S\{T\}$ and $\bar{R} = \bar{T}$ if $R = T$ then $\bar{R} = \bar{T}$

Table 6.1: Syntactic equivalence of formulas.

context or hole. The formula $S\{R\}$ is obtained by replacing the hole in $S\{\ \}$ by R. The curly braces are omitted when they are redundant, e.g., we shall write $S[R,T]$ instead of $S\{[R,T]\}$. A formula R is a *subformula* of a formula T if there is a context $S\{\ \}$ such that $S\{R\}$ is T.

Formulas are (syntactically) equivalent modulo the smallest equivalence relation induced by the equations shown in Table 6.1, where $\vec{R}$, $\vec{T}$ and $\vec{U}$ are finite sequences of formulas, and $\vec{T}$ is non-empty. Formulas are in *negation normal form* if negation occurs only over propositional variables. For example, the formulas $[a,b,c]$ and $(\vec{a},(\vec{b},\vec{c}))$ are equivalent: the first is not in negation normal form, the second is. Contrarily to the first, in the second formula, disjunction and conjunction only occur in their binary form.

The letters denoting formulas, i.e. S, P, Q, are schematic formulas. Likewise, $S\{\ \}$ is a schematic context. An inference rule ρ is a scheme written $\rho\frac{V}{U}$ where V and U are formulas that may contain schematic formulas and schematic formulas and schematic contexts. If neither U nor V contain a schematic context, then the inference rule is called *shallow*, otherwise it is called *deep*.

The inference rules of the symmetric system for propositional classical logic is shown is given in Table 6.2. It is called system SKS_g, where the first S stands for 'symmetric', K stands for 'klassisch' as in Gentzen's LK and the second S says that it is a system in the calculus of structures. Small letters are appended to the name of a system to denote variants. In this case, the g stands for 'general', meaning that rules are not restricted to atoms: they can be applied to arbitrary formulas.

The calculus of structures is *symmetric* in the sense that for each rule in the system, the dual rule is also in the system. The dual of an inference rule is obtained by exchanging premise and conclusion and replacing each connective by its De Morgan Dual.

The rules s, $w\!\downarrow$ and $c\!\downarrow$ are called respectively *switch*, *weakening* and *contraction*. Their dual rules carry the same name prefixed with a 'co-', so e.g. $w\!\uparrow$ is called co-weakening. Rules $i\!\downarrow$, $w\!\downarrow$, $c\!\downarrow$ are called down-rules and their duals are called up-rules. The dual of the switch rule s is the switch rule itself: it is *self-dual*. For example

$$\mathrm{c}\!\downarrow\dfrac{\mathrm{w}\!\uparrow\dfrac{[(a,\vec{b}),a]}{[a,a]}}{a} \qquad \text{is dual to} \qquad \mathrm{w}\!\downarrow\dfrac{\mathrm{c}\!\uparrow\dfrac{\vec{a}}{[a,a]}}{([\vec{a},b],\vec{a})}$$

down-rules		up-rules
$\mathrm{i}\!\downarrow\dfrac{S\{t\}}{S[R,\bar{R}]}$		$\mathrm{i}\!\uparrow\dfrac{S(R,\bar{R})}{S\{f\}}$
	$\mathrm{s}\dfrac{S([R,U],T)}{S[(R,T),U]}$	
$\mathrm{w}\!\downarrow\dfrac{S\{f\}}{S\{R\}}$		$\mathrm{w}\!\uparrow\dfrac{S\{R\}}{S\{t\}}$
$\mathrm{c}\!\downarrow\dfrac{S[R,R]}{S\{R\}}$		$\mathrm{c}\!\uparrow\dfrac{S\{R\}}{S(R,R)}$

Table 6.2: System SKS_g

6.1.2
Proof graphs for deep inference

Our proof-graphs introduced in Chapter 3, explore, basically, the subformula sharing and, with this facilitate, the normalization procedure elimination of maximal formulas. We propose directed graphs associated with SKS_g derivations, called deep-graphs, do not define a normalization procedure; however our graphs are a very convenient tool for defining and understanding several of its aspects. Our aim now is quickly provide the necessary notions about deep-graph.

Definition 29 L *is the union of the three sets of labels types:*

– R-Labels *is the set of inference labels:* $\{i\!\downarrow, i\!\uparrow, w\!\downarrow, w\!\uparrow, c\!\downarrow, c\!\uparrow, s, =^{c\vee}, =^{c\wedge}, =^{a\downarrow}, =^{a\uparrow}, =^{f\downarrow}, =^{t\downarrow}, =^{f\uparrow}, =^{f\wedge\downarrow}, =^{t\vee\downarrow}, =^{t\wedge\uparrow}, =^{f\wedge\uparrow}\}$

- F-Labels *is the set of formula labels:* $\{t,\ f\}$ *for units false and true, the letters* $\{a, b, c, ...\}$ *for atoms, and* $\{(.), [,]\}$ *for connectives ,*
- E-Labels *is the set of edge labels:* $\{l$ *(left),* r *(right),* p *(premise),* c *(conclusion)*$\}$*,*

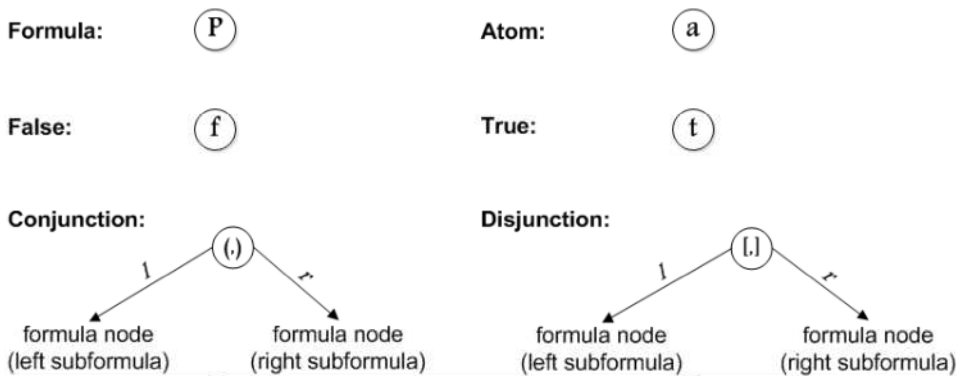


Figure 6.1: Formula nodes in Deep-graphs

Definition 30 *A* deep-graph G *is a directed graph* $\langle \mathrm{V},\ \mathrm{E},\ \mathrm{L} \rangle$ *where:* V *is a set of nodes,* L *is a set of labels,* E *is a set of labeled* edges $\langle v \in \mathrm{V},\ t \in \mathrm{L},\ v' \in \mathrm{V} \rangle$ *of source* v*, target* v' *and label* t *and is identified with the arrow* $v \xrightarrow{t} v'$*.*

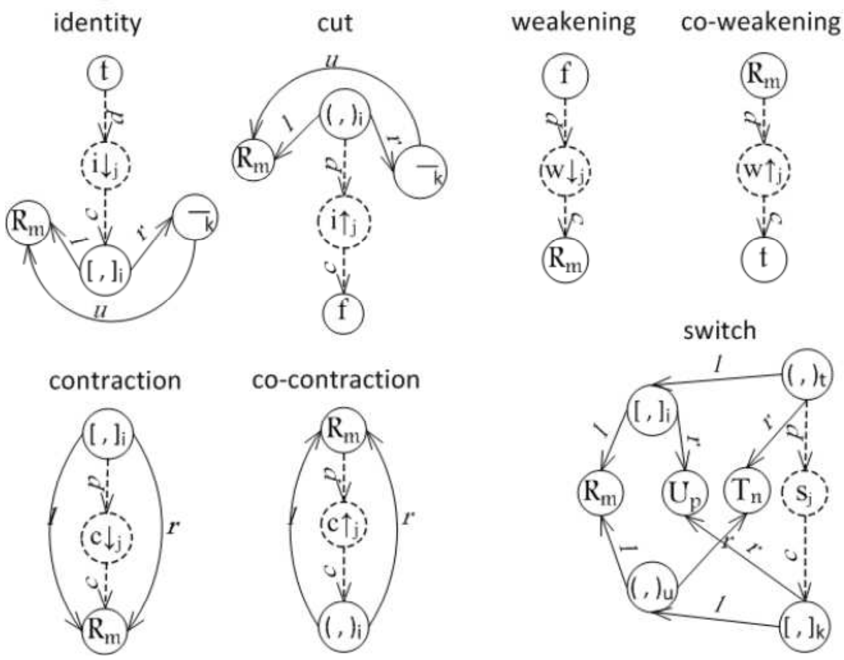
Deep-graphs are recursively defined as follows:

Basis *A formula graph* p *is a deep-graph.*

Rule *If* G_1 *is a formula graph with root node* R_m[1] *and* G_2 *is a deep-graph that contains a node* T_n *then the graph* G *that is defined as* $G_1 \oplus G_2$ *with one R-node* r_i *at the top position and the edges:* $R_m \xrightarrow{p_{new}} r_i$ *and* $r_i \xrightarrow{c_{new}} T_n$ *where* r_i *is one of the rules sketched below, is a deep-graph*

Structural Rules

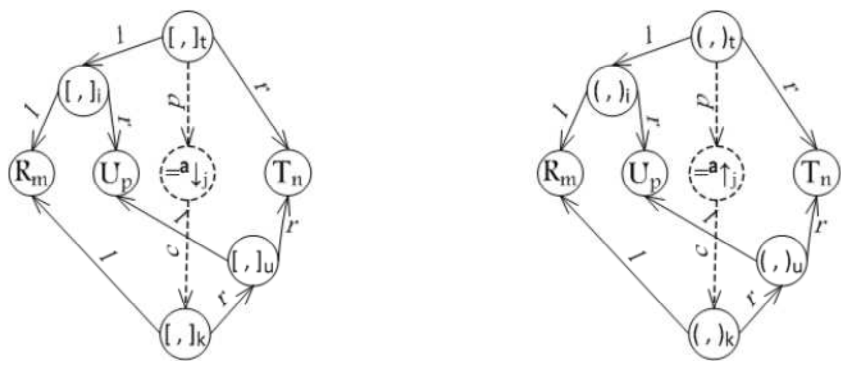
[1] We use the terms R_m, T_n and U_p to represent the principal connective of the formulas R, T and U, respectively.



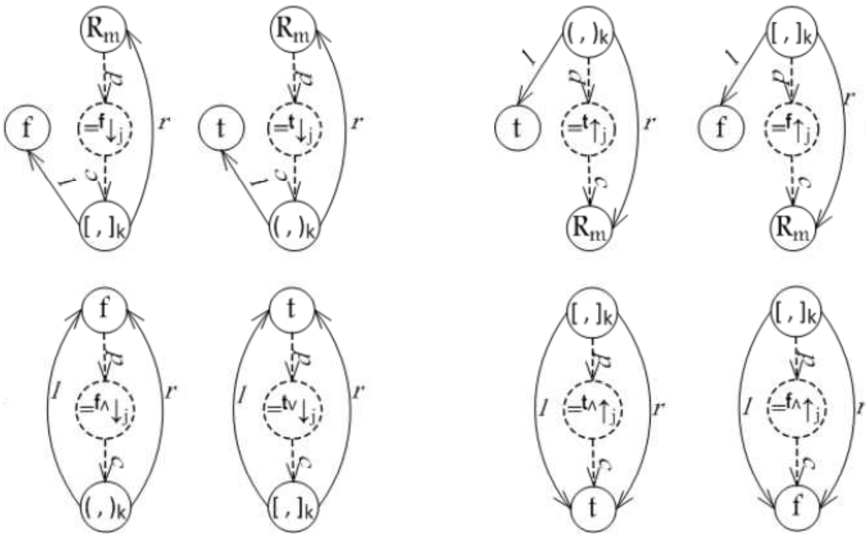
Logical Rules

Commutativity

Associativity



Units



6.1.3
Summary

We can say that deep-graph preserve the symmetry of Calculus of Structures: (i) all rules have one premise and one conclusion (vertical symmetry), (ii) there are dual rules, e.g. the identity rule and cut rule, weakening and co-weakening, (iii) the constant node f is symmetrical with t;

We intend, as future work, to propose a normalization procedure for deep-graphs, where we will use the technique of reduce cuts similar to what one does in normalization for propositional mimp-graphs.

6.2 Bi-intuitionistic logic seen from mimp-graph

6.2.1 A Brief review of bi-intuitionistic logic

Continuing with our aim of studying the complexity of proofs and provide more efficient theorem provers, we propose a proof-graph version for Bi-intuitionistic logic. We start with a brief overview of Bi-intuitionistic logic and then we present a proof-graph representation for this logic in the fragment composed by the implication and co-implication

Bi-intuitionistic logic is the extension of intuitionistic logic with the co-implication $\prec$ (also known as "subtraction" and "exclusion"), which is dual to implication $\rightarrow$, the formula $C \prec B$ is read as "B co-implies C" or as "C excludes B". Bi-intuitionistic logic can also be seen as the union of intuitionistic logic (lacking co-implication) with dual-intuitionistic logic (lacking implication).

Bi-intuitionistic logic was first studied by Rauszer as a Hilbert-style system and a sequent calculus (Rauszer 1974) (Rauszer 1977). In (Restall 1977), another sequent calculus is obtained by extending the multiple-conclusion sequent calculus for intuitionistic logic with co-implication rules dual to the implication rules, but it neither the sequent calculus of Rauszer are fully cut eliminable. Thus only cut-free calculi for Bi-intuitionistic logic either use extended sequent mechanisms such as labels (Pinto & Uustalu 2009), variables (Goré & Postniece 2010) or nested sequents (Goré, Postniece & Tiu 2010), or display calculi that rely on residuation (Goré 1998).

In this section we follow one kind of bi-intuitionistic propositional logic (*2Int*) recently conceived by Wansing (Wansing 2013) that combines a notion of dual proof (falsification) in addition to the more familiar notion of proof (verification) in Natural Deduction. A falsification of an implication $(A \rightarrow B)$ is a pair consisting of a verification of A and a falsification of B, whereas the verificationist must specify verification conditions for co-implications. Thus, Wansing proposed one single-conclusion system in natural deduction (*N2Int*), where introduction and elimination rules are dualized for intuitionistic propositional logic.

Definition 31 *Let ϕ be a denumerable set of atomic formulas. Elements from ϕ are denoted by p, q, r, p_1, p_2, ..., etc. Formulas generated from ϕ are be denoted by A, B, C, D, A_1, A_2, ..., etc. The propositional language* 2Int *is defined in Backus-Naur form as*

$$A ::= p_i \mid \bot \mid \top \mid (A \wedge A) \mid (A \vee A) \mid (A \rightarrow A) \mid (A \prec A).$$

In *2Int* $\bot$ is primitive, the co-negation $-A$ of A is defined as $(\top \prec A)$ and $\neg A$ of A is defined as $(A \to \bot)$.

Intuitionistic rule and its dual intuitionistic rule:

$$\begin{array}{c} \overline{\bot} \\ \hline A \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{\top}} \\ \hline\hline A \end{array} \qquad\qquad \begin{array}{c} \overline{A}\ \ \overline{B} \\ \hline A \wedge B \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{A}}\ \ \overline{\overline{B}} \\ \hline\hline A \vee B \end{array}$$

$$\begin{array}{c} \overline{A \wedge B} \\ \hline A \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{A \vee B}} \\ \hline\hline A \end{array} \qquad\qquad \begin{array}{c} \overline{A \wedge B} \\ \hline B \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{A \vee B}} \\ \hline\hline B \end{array}$$

$$\begin{array}{c} \overline{A} \\ \hline A \vee B \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{A}} \\ \hline\hline A \wedge B \end{array} \qquad\qquad \begin{array}{c} \overline{B} \\ \hline A \vee B \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{B}} \\ \hline\hline A \wedge B \end{array}$$

$$\begin{array}{c} \begin{array}{ccc} & [A] & [B] \\ & \vdots & \vdots \\ \overline{A \vee B} & \overline{C} & \overline{C} \end{array} \\ \hline C \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \begin{array}{ccc} & [\![A]\!] & [\![B]\!] \\ & \vdots & \vdots \\ \overline{\overline{A \wedge B}} & \overline{\overline{C}} & \overline{\overline{C}} \end{array} \\ \hline\hline C \end{array}$$

$$\begin{array}{c} [A] \\ \vdots \\ \overline{B} \\ \hline A \to B \end{array} \quad\rightsquigarrow\quad \begin{array}{c} [\![A]\!] \\ \vdots \\ \overline{\overline{B}} \\ \hline\hline B \prec A \end{array} \qquad\qquad \begin{array}{c} \overline{A}\ \ \overline{A \to B} \\ \hline B \end{array} \quad\rightsquigarrow\quad \begin{array}{c} \overline{\overline{A}}\ \ \overline{\overline{B \prec A}} \\ \hline\hline B \end{array}$$

Falsification of implications and verification of co-implications:

$$\begin{array}{c} \overline{A}\ \ \overline{\overline{B}} \\ \hline\hline A \to B \end{array} \qquad\qquad \begin{array}{c} \overline{\overline{A \to B}} \\ \hline A \end{array} \qquad\qquad \begin{array}{c} \overline{\overline{A \to B}} \\ \hline\hline B \end{array}$$

$$\begin{array}{c} \overline{A}\ \ \overline{\overline{B}} \\ \hline A \prec B \end{array} \qquad\qquad \begin{array}{c} \overline{A \prec B} \\ \hline A \end{array} \qquad\qquad \begin{array}{c} \overline{A \prec B} \\ \hline\hline B \end{array}$$

Table 6.3: N2Int: a inference system in Natural Deduction for 2Int

Table 6.3 gives the rules of the *N2Int*, where the introduction and elimination rules for intuitionistic propositional logic are dualized by replacing $\top$, $\bot$, $\wedge$, $\vee$, and $\to$ by their respective duals and single lines by double lines. Besides, we added the suggested rules by (Wansing 2013) for the falsification of implications and the verification of co-implications (see Table 6.3).

In a graphical presentation of derivations in *N2Int*, a single square brackets [ ] indicates the cancellation of an assumption (a formula taken to be true) and double-square brackets ⟦ ⟧ in order to indicate the cancellation of a counterassumption (a formula taken to be false).

6.2.2
Proof-graphs to bi-intuitionistic logic

We propose proof-graphs to bi-intuitionistic logic for fragment $\{\rightarrow, \prec\}$ in the mimp-graph style and we call it 2Int-graphs. In this way, 2Int-graphs are composed of the following objects:

1. The formula graphs that are composed by the formula nodes showed in Figure 6.2;

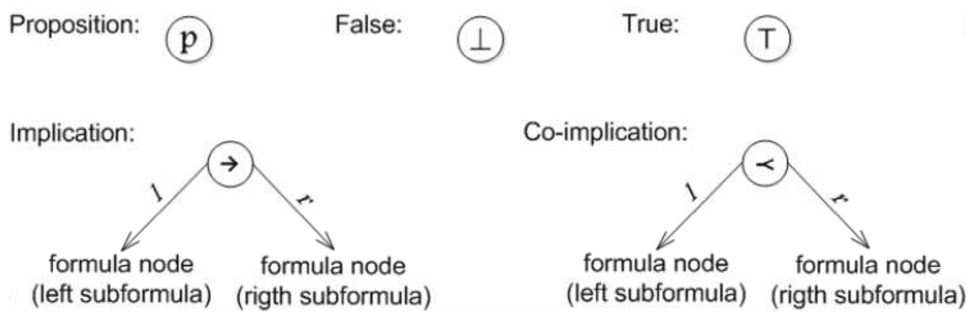
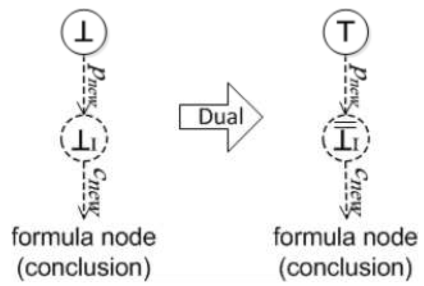


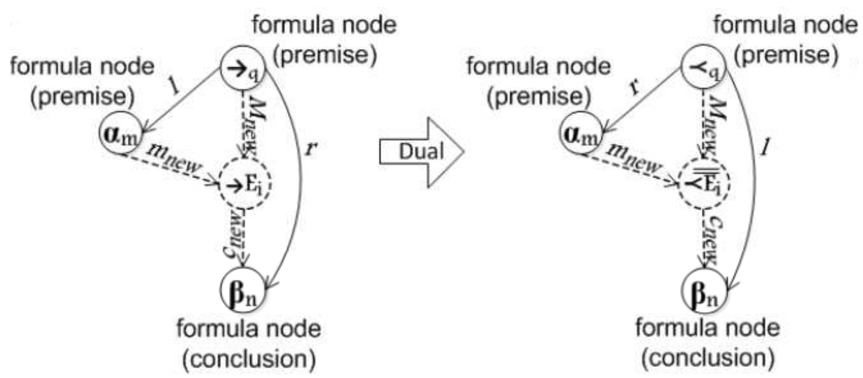
Figure 6.2: Formula nodes in 2Int-graph

2. A certain number of rules which are of the following types:

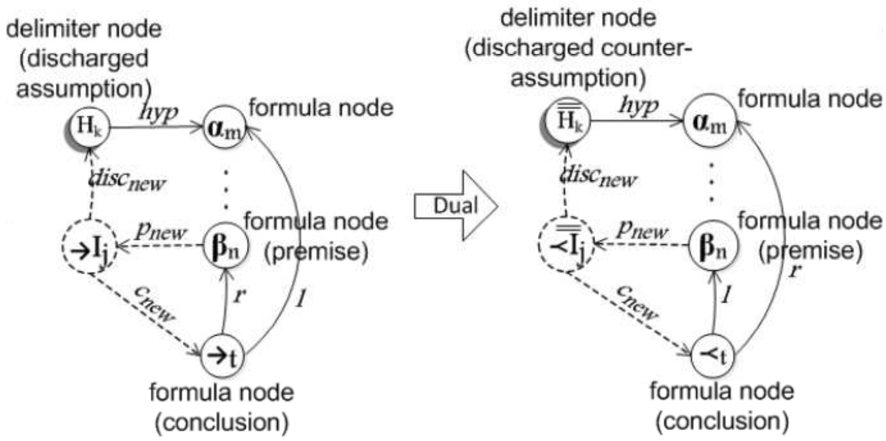
   Intuitionistic absurdity $\perp_I$ and its dual rule $\overline{\overline{\perp_I}}$.



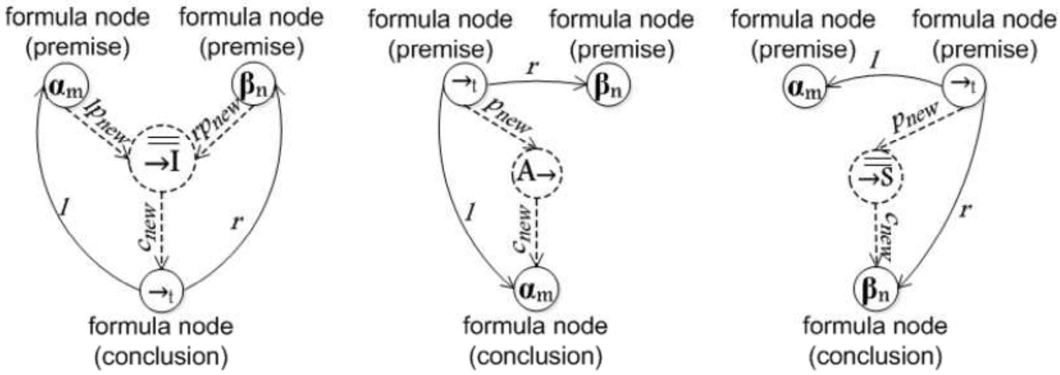
Implication Elimination $\rightarrow$E and its dual rule $\overline{\overline{\prec E}}$



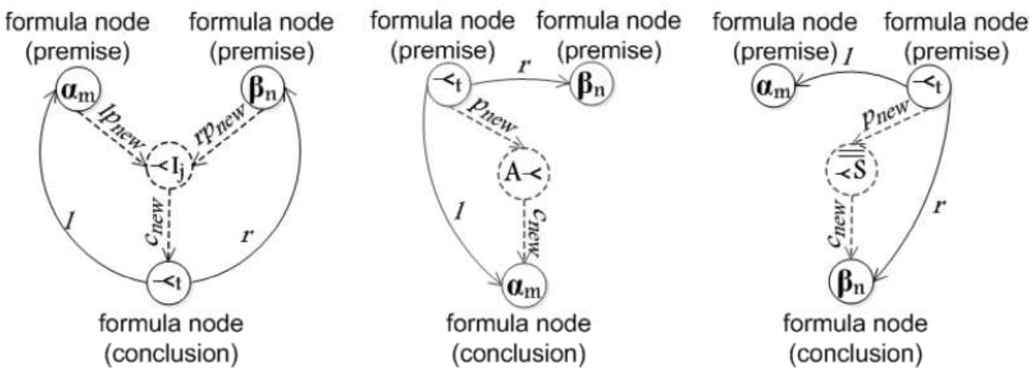
Implication Introduction $\rightarrow$I and its dual $\overline{\overline{\prec I}}$



$\overline{\overline{I}}$, $A \rightarrow$ and $\overline{\overline{S}}$ rules



$\prec$I, A$\prec$ and $\overline{\overline{\prec S}}$ rules



6.2.3
Examples

In the following figures we have examples of the translation of a derivation in *N2Int* to *2Int-graphs*.

$$\dfrac{\dfrac{\dfrac{}{\top} \quad \dfrac{[(B \to B) \prec A]^1}{A}}{\top \prec A}}{((B \to B) \prec A) \to (\top \prec A)}\,1 \qquad \Rightarrow$$

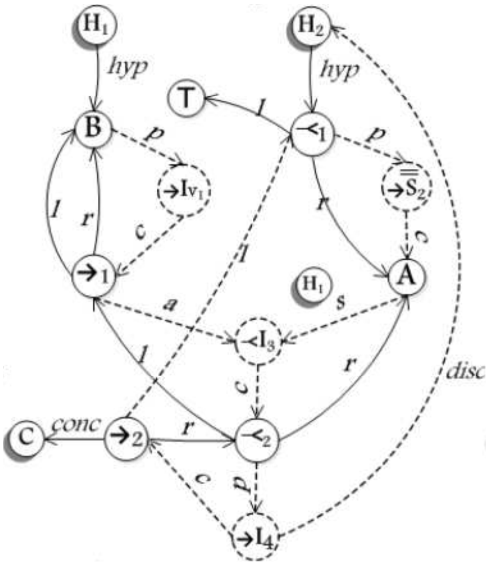


Figure 6.3: Translation of derivation in *N2Int* to *2Int-graph*

$$\dfrac{\dfrac{\dfrac{B}{B \to B} \quad \dfrac{[\top \prec A]^1}{A}}{(B \to B) \prec A}}{((\top \prec A) \to ((B \to B) \prec A)}\,1 \qquad \Rightarrow$$

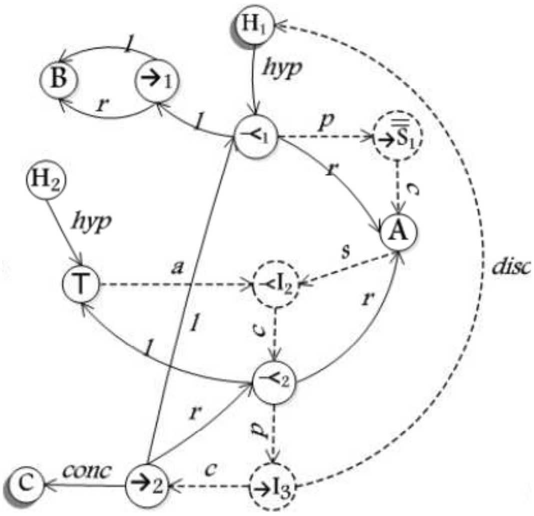


Figure 6.4: Translation of derivation in *N2Int* to *2Int-graph*

6.2.4
Observation

The application of a mimp-style representation to Bi-intuitionistic logic (2Int) aims to verify that using this graph representation results in a reduced size of the proofs with respect to traditional ways of presentation in Natural

Deduction (N2Int). It also allows a better understanding of the proving process, due to the intuitive graphical interpretation the graphs provide. For example, the use of the delimiter node hypothesis (assumptions and counter-assumptions) has also proven useful. In particular, it has made duality identification manageable and more elegant, in such way semantic properties of logical connectives are determined by the rules nodes.