Pontifícia Universidade Católica
do Rio de Janeiro

Marcela Quispe Cruz

Some Results in a Proof-theory Based on Graphs

Tese de Doutorado

Thesis presented to the Programa de Pós Graduação em Informática of the Departamento de Informática, PUC-Rio as partial fulfillment of the requirements for the degree of Doutor em Informática

Advisor : Prof. Edward Hermann Haeusler
Co–Advisor: Prof. Lew Gordeev

Rio de Janeiro
November 2014

Marcela Quispe Cruz

Some Results in a Proof-theory Based on Graphs

Thesis presented to the Programa de Pós Graduação em Informática of the Departamento de Informática, PUC-Rio as partial fulfillment of the requirements for the degree of Doutor em Informática. Approved by the following commission:

Rio de Janeiro, November 13th, 2014

Marcela Quispe Cruz

Marcela Quispe Cruz graduated from the Universidad Nacional de San Agustín (UNSA – Arequipa, Peru) in Computer Science. She finished her master's thesis at the Universidade Federal de Pernambuco (Pernambuco, Brazil) in Computer Science. She then obtained a D.Sc. at the Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio – Rio de Janeiro, Brazil) in Informatics (Theory of Computation).

Bibliographic data

Quispe, Marcela

Some Results in a Proof-theory Based on Graphs / Marcela Quispe Cruz ; advisor: Edward Hermann Haeusler; co–advisor: Lew Gordeev. — 2014.
90 f. : il. ; 30 cm

Tese (Doutorado em Informática)-Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2014.
Inclui bibliografia

1. Informática – Teses. 2. Teoria da Prova. 3. Grafos de Prova. 4. Dedução Natural. 5. Complexidade de Provas. 6. Normalização Forte. 7. Lógica Clássica. 8. Lógica Minimal Implicacional. 9. Inferência Profunda. 10. Lógica Bi-intuicionista. I. Haeusler, Edward Hermann. II. Gordeev, Lew. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

Acknowledgments

To God for keeping me in your tender loving care.

To my advisor, Prof. Edward Hermann Haeusler, for his encouragement, patience, caring, expert advice and friendship. To Prof. Lew Gordeev who also advised me for his useful suggestions.

To my professor Luiz Carlos Pereira for his enthusiasm to teach.

To professors Anjolina Grisi de Oliveira, Marco Antonio Casanova, Mauricio Ayala Rincón and Alex de Vasconcellos Garcia for their valuable comments and suggestions.

To my colleagues of TECMF, Cecilia, Bruno, Jefferson and Hugo.

To my friends in Rio de Janeiro for their companionship.

To the secretaries of the postgraduate program for their kind assistance.

To my parents Vicente and Mercedes, my brother Helbert and my sister Nancy, for all of their comprehension and kindness during this process.

To the CAPES for the financial support without which this work could not have been accomplished. To everybody who, directly or not, contributed in this journey, thank you.

Abstract

Quispe, Marcela; Haeusler, Edward Hermann (advisor); Gordeev, Lew (co-advisor). **Some Results in a Proof-theory Based on Graphs**. Rio de Janeiro, 2014. 90p. D.Sc. Thesis — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Traditional proof theory of Propositional Logic deals with proofs which size can be huge. Proof theoretical studies discovered exponential gaps between normal or cut free proofs and their respective non-normal proofs. Thus, the use of proof-graphs, instead of trees or lists, for representing proofs is getting popular among proof-theoreticians. Proof-graphs serve as a way to provide a better symmetry to the semantics of proofs and a way to study complexity of propositional proofs and to provide more efficient theorem provers, concerning size of propositional proofs. The aim of this work is to reduce the weight/size of deductions. We present formalisms of proof-graphs that are intended to capture the logical structure of a deduction and a way to facilitate the visualization. The advantage of these formalisms is that formulas and sub-deductions in Natural Deduction, preserved in the graph structure, can be shared deleting unnecessary sub-deductions resulting in the reduced proof. In this work, we give a precise definition of proof-graphs for purely implicational logic, then we extend this result to full propositional logic and show how to reduce (eliminating maximal formulas) these representations such that a normalization theorem can be proved by counting the number of maximal formulas in the original derivation. The strong normalization will be a direct consequence of such normalization, since that any reduction decreases the corresponding measures of derivation complexity. Continuing with our aim of studying the complexity of proofs, the current approach also give graph representations for first order logic, deep inference and bi-intuitionistic logic.

Keywords

Proof Theory; Proof-graphs; Natural Deduction; Proof Complexity; Strong Normalization; Classical Logic; Implicational Minimal Logic; Deep Inference; Bi-intuitionistic Logic;

Resumo

Quispe, Marcela; Haeusler, Edward Hermann (orientador); Gordeev, Lew (co-orientador). **Alguns resultados em teoria de prova baseado em grafos**. Rio de Janeiro, 2014. 90p. Tese de Doutorado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A teoria da prova tradicional da lógica proposicional trata provas cujos tamanhos podem ser demasiado grandes. Estudos teóricos de prova descobriram diferenças exponenciais entre provas normais ou livres de corte e suas respectivas provas não-normais. Assim, o uso de grafos-de-prova, ao invés de árvores ou listas, para representar provas está se tornando mais popular entre teóricos da prova. Os grafos-de-prova servem como uma forma de proporcionar uma melhor simetria para a semântica de provas e uma maneira de estudar a complexidade das provas proposicionais e fornecer provadores de teoremas mais eficientes, em relação ao tamanho das provas proposicionais. O objetivo deste trabalho é reduzir o peso/tamanho de deduções. Apresentamos formalismos de grafos de prova que visam capturar a estrutura lógica de uma dedução e uma forma de facilitar a visualização das propriedades. A vantagem destes formalismos é que as fórmulas e sub-deduções em dedução natural, preservadas na estrutura de grafo, podem ser compartilhadas eliminando sub-deduções desnecessárias resultando na prova reduzida. Neste trabalho, damos uma definição precisa de grafos de prova para a lógica puramente implicacional, logo estendemos esse resultado para a lógica proposicional completa e mostramos como reduzir (eliminando fórmulas máximas) essas representações de tal forma que um teorema de normalização pode ser provado através da contagem do número de fórmulas máximas na derivação original. A normalização forte será uma consequência direta desta normalização, uma vez que qualquer redução diminui as medidas correspondentes da complexidade da derivação. Continuando com o nosso objetivo de estudar a complexidade das provas, a abordagem atual também fornece representações de grafo para lógica de primeira ordem, a inferência profunda e lógica bi-intuitionista.

Palavras-chave

Teoria da Prova; Grafos de Prova; Dedução Natural; Complexidade de Provas; Normalização Forte; Lógica Clássica; Lógica Minimal Implicacional; Inferência Profunda; Lógica Bi-intuicionista;

Contents

List of Figures

1 Introduction

The use of proof-graphs, instead of trees or lists, for representing proofs is getting popular among proof-theoreticians. Proof-graphs serve as a way to provide a better symmetry to the semantics of proofs (Oliveira & Queiroz 2003) and a way to study complexity of propositional proofs and to provide more efficient theorem provers, concerning size of propositional proofs. In (Bonet & Buss 1993), one can find a complexity analysis of the size of Frege systems, Natural Deduction systems and Sequent Calculus concerning their tree-like and list-like representation. This leads to $O(nlog(n))$ improvement in the size of the list-based proofs compared to tree-like proofs, which is based on the observation that the hypotheses occur only once in the lists and more than once in the trees. Thus sharing formulas helps to reduce the size of proofs. There are related works, e.g. (Alves, Fernández & Mackie 2011), that use graphs for representing proofs, pointing out that proof-graphs offer a better way to facilitate the visualisation and understanding of proofs in the underlying logic.

On the other hand (Finger 2005), (da Costa 2007) and (Gordeev, Haeusler & Costa 2009) show that the use of Directed Acyclic Graphs (DAGs) together with mechanisms of unification/substitution in proof representations has compacting/compressing factor equivalent to cut-introduction. And, obviously, graphs can save space by means of reference, instead of plain copying. This work shows yet another advantage of using graphs for representing proofs. First, we show that using "mixed" graph representations of formulas and inferences in Natural Deduction in the purely implicational minimal logic one can obtain a (weak) normalization theorem that, in fact, is a strong normalization theorem. Moreover the corresponding normalization procedure does not exceed the size of the input, which sharply contrasts to the well-known exponential speed-up of standard normalization. The choice of purely implicational minimal logic ($M^{\rightarrow}$) is motivated by the fact that the computational complexity of the validity of $M^{\rightarrow}$ is PSPACE-complete and can polynomially simulate classical, intuitionistic and full minimal logic (Statman 1979) as well as any propositional logic with a Natural Deduction system satisfying the subformula property (Haeusler 2013). Then we extend this result to propositional logic obtaining also strong normalization.

In a more general context, this work has been conducted as part of a bigger tree-to-graph proof compressing research project. The purpose of such proof compression is:

1. To construct small (if possible, minimal) graph-like representations of standard tree-like proofs in a given proof system and – in the propositional case – investigate the corresponding short graph-like theorem provers.
2. To find short (say, polynomial-size) graph-like analogous of standard tree-like proof theoretic operations like e.g. normalization in Natural Deduction and/or cut-elimination in Sequent Calculus.

Note that the present work fulfills both conditions with regard to the mimp-graph representation (see below) of chosen Natural Deduction and the corresponding notion of formula-minimality (see Theorems 3, **??**, 4 and 5).

Back to the proof normalization, recall the following properties of a given structural deductive system (Natural Deduction, Sequent Calculus, etc):

– Normal form: To each derivation of α from Δ there is a normal derivation of α from $\Delta' \subseteq \Delta$.

– Normalization: To each derivation of α from Δ there is a normal derivation of α from $\Delta' \subseteq \Delta$, obtained by a particular strategy of reductions application.

– Strong Normalization: To each derivation of α from Δ there is a normal derivation of α from $\Delta' \subseteq \Delta$. This normal form can be obtained by applying reductions to the original derivation in any ordering.

The strong normalization property for a natural deduction system is usually proved by the so-called semantical method:

– Define a property $P(\pi)$ on derivations π in the Natural Deduction system;

– Prove that this property implies strong normalization, that is $\forall\pi(P(\pi) \rightarrow SN(\pi))$, where $SN(X)$ means that X is strongly normalizable;

– Prove that $\forall\pi P(\pi)$.

There are well-known examples of this property $P(X)$: (1) Prawitz's "strong validity"; (2) Tait's "convertibility"; (3) Jervell's "regularity"; (4)

Leivant's "stability"; (5) Martin-Löf's "computability"; (6) Girard's "candidate de reducibilité". Note that such semantical method to prove strong normalization is unconstructive and even in the case of purely implicational fragment of minimal logic it provides no combinatorial insight into the nature of strong normalization. Another, more constructive strategy would be to show that there is a worst sequence of reductions always produces a normal derivation. Let us call it a syntactic method of proving the strong normalization theorem. The method used in the present research is that any sequence of reductions always produces a normal derivation. This means that the order in which cuts (maximal formulas) are eliminated has no impact on the end-result. This is obtained by brute force: the proof consists of an exhaustive case-analysis.

Other methods use assignments of rather complicated measures to derivations such that arbitrary reductions decrease the measure, which by standard inductive arguments yields a desired proof of the strong normalization. In this thesis we show how to represent derivations in a graph-like form to $M^{\rightarrow}$ and full propositional logic, and how to reduce (eliminating maximal formulas) these representations such that a normalization theorem can be proved by counting the number of maximal formulas in the original derivation. The strong normalization is a direct consequence of such normalization, since any reduction decreases the corresponding measure of derivation complexity. The underlying intuition comes from the fact that our graph representations use only one node for any two identical formulas occurring in the original Natural Deduction derivation (see Theorem 3 for a more precise description).

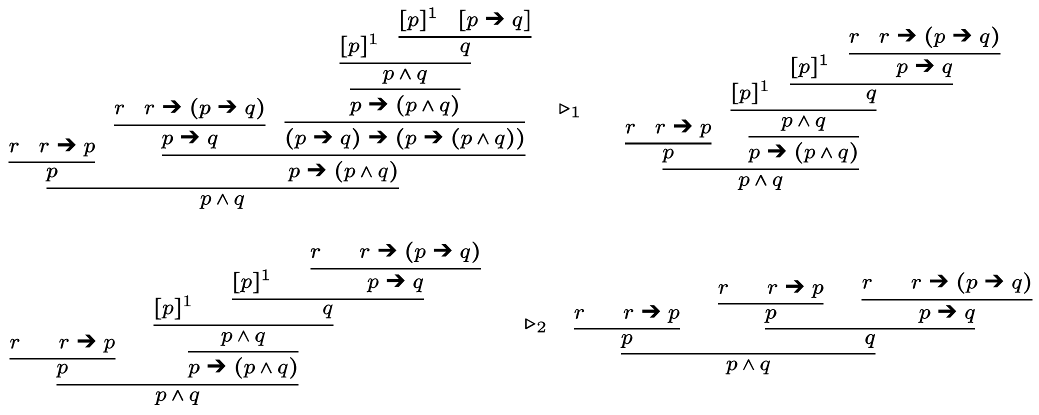


Figure 1.1: Example of derivation with two steps of reduction.

We show in Figure 1.1 an example of the eliminating a maximal formula in a derivation in Natural Deduction. The formula $p \rightarrow (p \wedge q)$ is not a maximal formula before a reduction ($\rhd_1$) is applied to eliminate the maximal formula $(p \rightarrow q) \rightarrow (p \rightarrow (p \wedge q))$. This possibility of having hidden maximal formulas in

ND is the main reason to use more sophisticated methods whenever proving strong normalization.

In Figure 1.2 we show an embedding of this derivation into a mimp-graph. This example shows the reason why our normalization procedure is directly a strong normalization. We remark that there is no possibility to hide a maximal formula because all formulas are represented only once in the graph (see Figure 1.2). In this graph $p \rightarrow (p \wedge q)$ is already a maximal formula. We can choose to remove any of the maximal formulas. If $p \rightarrow (p \wedge q)$ is chosen to be eliminated, by the mimp-graph elimination procedure, its reduction eliminates the $(p \rightarrow q) \rightarrow (p \rightarrow (p \wedge q))$ too. On the other hand, the choice of $(p \rightarrow q) \rightarrow (p \rightarrow (p \wedge q))$ to be reduced only eliminates itself. In any case the number of maximal formulas decreases and the mimp-graph becomes as shown in Figure 1.3.

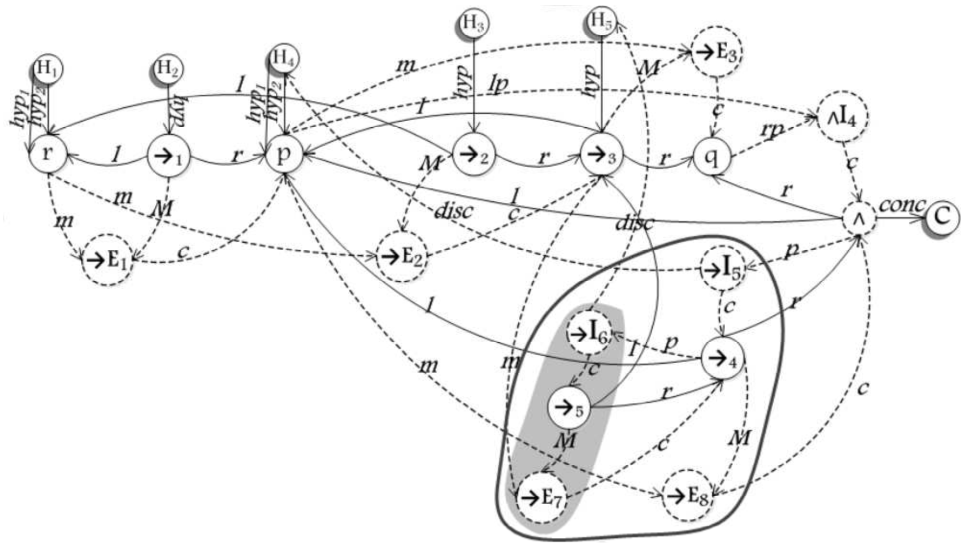


Figure 1.2: Mimp-graph translation of derivation in Figure 1.1.

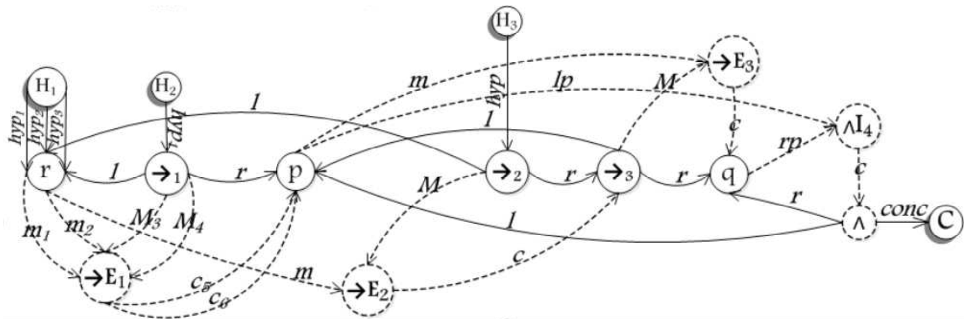


Figure 1.3: Normalized mimp-graph of the example in Figure 1.2.

Continuing with our aim of studying the complexity of proofs, the current approach also give graph representations for first order logic, deep inference and bi-intuitionistic logic.

1.0.1 Related work

The related idea of proof-graphs has been investigated in the last decade, for classical and intuitionistic logic. More recently, N-graphs introduced by de Oliveira (Oliveira & Queiroz 2003) is a proof system originally developed for classical logic, as a suitable solution to the lack of symmetry in classical ND logic. Mainly because N-Graphs use a multiple conclusion proof structure. N-Graphs have also been adapted for intuitionistic logic (Quispe-Cruz, de Oliveira, de Queiroz & de Paiva 2014).

In (Geuvers & Loeb 2007) another approach to represent Natural Deduction using graphs is proposed. It reports a graph-representation of Natural Deduction, in Gentzen as well as Fitch's style. In fact the proofs are represented as hypergraphs, or boxed-graphs, with possibility of sharing subproofs. It is developed not only for the implicational fragment, although the representation of linear logic proofs is related as further work. Our approach is different from (Geuvers & Loeb 2007) in that we include graph-representations of formulas in the proofs. The fact that our normalization procedure leads to strong-normalization is a consequence of sharing subformulas, and hence subproofs, in our proof-graph representations. It is unclear whether a similar result is available using (Geuvers & Loeb 2007).

Other previous research concerning the use of graphs to represent proofs was developed on connections to substructural logics as Linear Logic, see (Girard 1996) and (Girard, Lafont & Regnier 1995) for example. The main motivation of this just mentioned investigations is to provide a sound way of representing Linear Logic proofs without dealing with unique labeling and complicated rules for relabeling and discharging mechanisms need to represent Linear Logic proofs as trees in Natural Deduction style as well as in Sequent Calculus.

Proof-nets were such representations and a syntactical criteria on the possible paths on them were considered as a soundness criteria for a proof-graph to be a proof-net. Proof-nets have a cut-rule quite similar to the cut in Sequent Calculus. For the Multiplicative fragment of Classical Linear Logic, there is a linear time cut-elimination theorem. However, when the additive versions of the connectives are considered, the usual complexity of the cut-elimination raises up again. Linear Logic is an important Logic whenever we consider the study of a concurrent computational system and semantics of it strongly uses concurrency theory concepts. Our investigation, on the other hand, is not motivated by proof-theoretical semantics [1]. From the purely proof-

[1] The name that nowadays is used to denote the kind of research pioneered by Jean-Yves

theoretical point of view, we use graphs to reduce the redundancy in proofs in such a way that we do not allow hidden maximal formulas in our graph representation of a Natural Deduction proof.

Finally, we mention Alessio-Gundersen (Guglielmi & Gundersen 2008) work, where a kind of flow graph is used to present an abstract graphical framework capable of representing the calculus of structures in deep inference and is also presented a normalization mechanism via an abstract graphical framework for SKS system.

1.1 Organization

Chapter 2 gives the basic notions of natural deduction that is considered in the thesis.

Chapter 3 introduces our proof-graphs for minimal implicational propositional logic.

Chapter 4 proposes proof-graphs with explicit sharing of sub-proofs by means of boxes to border the set of shared rules.

Chapter 5 adds the full minimal propositional logic to the mimp-graph formalism (propositional mimp-graph) with its normalization procedure and then adds a version for first order logic and the set of transformations that we need to describe the normalization.

Chapter 6 starts with a brief overview of deep inference and the *calculus of structures* by Guglielmi (Guglielmi 2007) and then presents a proof-graph representation for this calculus; and Section 6.2 introduces the Bi-intuitionistic Logic and then shows how the formalism N2Int can be embedded in proof-graphs.

2 Background

In this chapter, we present the basic notions related to the logics that will be considered in the thesis. We will start by introducing natural deduction that will serve as a basis for developing proof-graphs. Thus, we review the first order logic formulation in Gentzen-Prawitz' Natural Deduction.

2.1 Natural Deduction

2.1.1 A Natural Deduction System

The natural deduction (ND) is formulated in a language with logical symbols: connectives $\wedge$, $\vee$, $\rightarrow$, the quantifiers $\forall$ and $\exists$, parentheses, bound variables $x, y, z, ...$, free variables $a, b, c, ...$ and the symbol for contradiction, $\bot$(absurdity). Non-logical symbols: the propositional letters, predicates $P, Q, R, ...$, functional symbols. The set of formulas $A, B, C, ...$ can be generated with these logical symbols and non logical symbols. Negation of a formula A is expressed by $A \rightarrow \bot$.

Definition 1

- ***Minimal Implicational Logic.*** *It is a version of classical propositional calculus: Formulas are built only with propositional letters and the implication connective* $\rightarrow$*. There are only two simple inference rules in ND: the implication introduction (*$\rightarrow$I*) and the implication elimination (*$\rightarrow$E*) that we will see later.*
- ***Propositional Logic.*** *We will now add to propositional logic, the connectives: conjunction (*$\wedge$*) and disjunction (*$\vee$*) and their introduction and elimination rules.*
- ***First Order Logic.*** *In addition to the symbols of propositional logic (connectives, propositional letters and parentheses), It includes the quantifiers* $\forall$ *and* $\exists$*, predicates, functional symbols and constants.*

An inference rule is a scheme written as:

$$\frac{S_1, S_2, ..., S_n}{S}\,\rho\ ,$$

where ρ is the name of the rule, $S_1, S_2, ..., S_n$ are premise formulas and S is the conclusion formula. The basic idea of Natural Deduction is an *asymmetry*: Deductions take the form of tree-like structure, starting with one or more hypotheses as the leaves but having only one conclusion as the root.

The natural deduction system is described by means of introduction and elimination rules for each connective/quantifier plus the intuitionistic absurdity rule ($\bot_I$) and the classical absurdity rule ($\bot_C$). These inference rules are as follows (Menezes & Haeusler 2006):

Conjunction

$$\dfrac{\begin{array}{cc}\Pi_1 & \Pi_2\\ A & B\end{array}}{A \wedge B}\,\wedge\mathrm{I} \qquad \dfrac{\begin{array}{c}\Pi_1\\ A \wedge B\end{array}}{A}\,\wedge\mathrm{E}_R \qquad \dfrac{\begin{array}{c}\Pi_1\\ A \wedge B\end{array}}{B}\,\wedge\mathrm{E}_L$$

Disjunction

$$\dfrac{\begin{array}{c}\Pi_1\\ A\end{array}}{A \vee B}\,\vee\mathrm{I}_R \qquad \dfrac{\begin{array}{c}\Pi_1\\ B\end{array}}{A \vee B}\,\vee\mathrm{I}_L \qquad \dfrac{\begin{array}{ccc} & [A]^u & [B]^v\\ \Pi_1 & \Pi_2 & \Pi_3\\ A \vee B & C & C\end{array}}{C}\,(\vee\mathrm{E}, u, v)$$

Implication

$$\dfrac{\begin{array}{c}[A]^u\\ \Pi_1\\ B\end{array}}{A \rightarrow B}\,(\rightarrow\mathrm{I},\ u) \qquad \dfrac{\begin{array}{cc}\Pi_1 & \Pi_2\\ A & A \rightarrow B\end{array}}{B}\,\rightarrow\mathrm{E}$$

Existential Quantifier

$$\dfrac{A(t)}{\exists x A(x)}\,\exists\mathrm{I} \qquad \dfrac{\begin{array}{cc} & [A(a)]\\ & \vdots\\ \exists x A(x) & C\end{array}}{C}\,\exists\mathrm{E}$$

Universal Quantifier

$$\dfrac{A(a)}{\forall x A(x)}\,\forall\mathrm{I} \qquad \dfrac{\forall x A(x)}{A(t)}\,\forall\mathrm{E}$$

Intuitionistic and Classical Absurdity

$$\dfrac{\begin{array}{c}\Pi_1\\ \bot\end{array}}{A}\,\bot_I \qquad \dfrac{\begin{array}{c}[A \rightarrow \bot]^u\\ \Pi_1\\ \bot\end{array}}{\mathrm{A}}\,(\bot_C, u)$$

Some rules allow to discharge the hypotheses, when a formula is inferred, it becomes independent of a certain hypothesis. We denote discharged hypothesis by using square brackets and use an index to relate the hypothesis to the rule application that discharges it.

- In $\forall$I-rule the parameter a can not occur in any hypothesis on which the proof of $A(a)$ depends.
- In $\exists$E-rule the parameter a can not occur neither in $\exists x A(x)$, nor in C, nor in any hypothesis on which the upper occurrence of C depends other than $A(a)$.

Figure 2.1 shows a proof in Natural Deduction. The formulas $\forall y F(a,y)$ and $\exists x \forall y F(x,y)$ are assumed at first. Then they are discharged at the steps $\exists E_1$ and $\to I_2$ respectively.

$$\dfrac{\dfrac{[\exists x\forall y F(x,y)]^u \quad \dfrac{\dfrac{\dfrac{[\forall y F(a,y)]^v}{F(a,b)}\,\forall E}{\exists x F(x,b)}\,\exists I}{\forall y\exists x F(x,y)}\,\forall I}{\forall y\exists x F(x,y)}\,(\exists E,\ v)}{\exists x\forall y F(x,y) \to \forall y \exists x F(x,y)}\,(\to I, u)$$

Figure 2.1: A proof in Natural Deduction.

In the definition of proof-graphs for first order logic, we will only be concerned with natural deduction for intuitionistic logic, that is obtained by the set of rules for ND, excluding the classical absurdity rule ($\bot_c$).

2.1.2
Normalization

In some kinds of derivations, we have the existence of redundancies. Specifically, we have redundancies or "detours" like introduction followed by elimination of a connective that can be transformed into a proof without the two rules. Proof transformations by eliminating redundancies is called the normalization procedure, which is the main computational interest for us and was introduced by Gentzen and later on developed much further by Prawitz, who considerably extent Gentzen' techniques and results. For an extensive treatment see (van Dalen 1994).

Definition 2 *A formula occurrence γ is a* maximal formula *in a derivation when it is the conclusion of an introduction rule and the major premise of an elimination rule. γ is called a maximal formula.*

Derivations will systematically be converted into simpler ones by "elimination of maximal formulas". The main cases are defined below:

Definition 3 (One Step Reduction)

- $\wedge$I *followed by* $\wedge\mathrm{E}_i$*:*

$$\dfrac{\dfrac{\begin{matrix}\Pi_1\\ A_1\end{matrix}\qquad\begin{matrix}\Pi_2\\ A_2\end{matrix}}{A_1\wedge A_2}\,\wedge\mathrm{I}}{A_i}\,\wedge\mathrm{E}_i \qquad \textit{converts to} \qquad \begin{matrix}\Pi_i\\ A_i\end{matrix}$$

- $\rightarrow$I *followed by* $\rightarrow$E*:*

$$\dfrac{\begin{matrix}\Pi_1\\ A\end{matrix}\qquad \dfrac{\begin{matrix}[A]^u\\ \Pi_2\\ B\end{matrix}}{A\rightarrow B}\,(\rightarrow\mathrm{I},\ u)}{B}\,\rightarrow\mathrm{E} \qquad \textit{converts to} \qquad \begin{matrix}\Pi_1\\ [A]\\ \Pi_2\\ B\end{matrix}$$

- $\vee$I *followed by* $\vee$E*:*

$$\dfrac{\dfrac{\begin{matrix}\Pi\\ A_i\end{matrix}}{A_1\vee A_2}\,\vee\mathrm{I}\qquad\begin{matrix}[A_1]^u\\ \Pi_1\\ B\end{matrix}\qquad\begin{matrix}[A_2]^v\\ \Pi_2\\ B\end{matrix}}{B}\,(\vee\mathrm{E},u,v) \qquad \textit{converts to} \qquad \begin{matrix}\Pi\\ [A]\\ \Pi_i\\ B\end{matrix}$$

- $\forall$I *followed by* $\forall$E*:*

$$\dfrac{\dfrac{\begin{matrix}\Pi\\ A\end{matrix}}{\forall xA[x/y]}\,\forall\mathrm{I}}{A[t/y]}\,\forall\mathrm{E} \qquad \textit{converts to} \qquad \begin{matrix}\Pi[t/y]\\ A[t/y]\end{matrix}$$

- $\exists$I *followed by* $\exists$E*:*

$$\dfrac{\dfrac{\begin{matrix}\Pi_1\\ A(t)\end{matrix}}{\exists xA(x)}\,\exists\mathrm{I}\qquad\begin{matrix}[A(y)]^u\\ \Pi_2\\ B\end{matrix}}{B}\,(\exists\mathrm{E},u) \qquad \textit{converts to} \qquad \begin{matrix}\Pi_1\\ A(t)\\ \Pi_2[t/y]\\ B\end{matrix}$$

Notation $\Pi >_1 \Pi'$ stands for "Π is converted to Π' ". $\Pi > \Pi'$ stands for "there is a finite sequence of conversions $\Pi = \Pi_0 >_1 \Pi_1 >_1 \dots >_1 \Pi_{n-1} = \Pi'$ " and $\Pi \geq \Pi'$ stands for $\Pi > \Pi'$ or $\Pi = \Pi'$. (Π reduces to Π').

Definition 4 *If there is no* Π_1' *such that* $\Pi_1 >_1 \Pi_1'$ *(i.e. if* Π_1 *does not contain maximal formulas), then we call* Π_1 *a* normal derivation*, or we say that* Π_1 *is in* normal form*, and if* $\Pi \geq \Pi'$ *where* Π' *is normal, then we say that* Π normalizes *to* Π'*.*

We say that $>$ has the *weak normalization property* if every derivation normalizes. Moreover, when this property holds independently of the strategy that is used in applying the reduction steps, one says that the system satisfies the *strong normalization property.*

Theorem 1 (Weak Normalization) *All derivations normalise.*

Theorem 2 (Subformula Principle) *Every formula occurrence in a normal deduction of A from Γ has the shape of a subformula of A or of some formula of Γ, except for hypotheses discharged by applications of the $\bot_C$-rule and for occurrences of $\bot$ that stand immediately below such hypotheses.*

3
Mimp-graphs: Graphs for Minimal Implicational Logic

In this chapter we define graph representations using "mixed" formulas and inferences in Natural Deduction in the purely implicational minimal logic; then we obtain a (weak) normalization theorem that, in fact, is a strong normalization theorem. The choice of purely implicational minimal logic ($M^{\to}$) is motivated by the fact that the computational complexity of the validity of $M^{\to}$ is PSPACE-complete and can polynomially simulate classical, intuitionistic and full minimal logic (Statman 1979) as well as any propositional logic with a Natural Deduction system satisfying the subformula property (Haeusler 2013).

3.1
Definition of Mimp-graphs

Mimp-graphs are special directed graphs whose nodes and edges are assigned with labels. Moreover we distinguish two parts, one representing the inferences of a proof, and the other the formulas. For the formula-part of a mimp-graph, we use formula graphs as a basis and consist only of formula nodes (see Definition 8). A formula can also be presented as a formula tree, but sharing within a formula is possible with formula graphs. An example is shown in the Figure 3.1: the propositions p and q each only need to occur once in the graph.

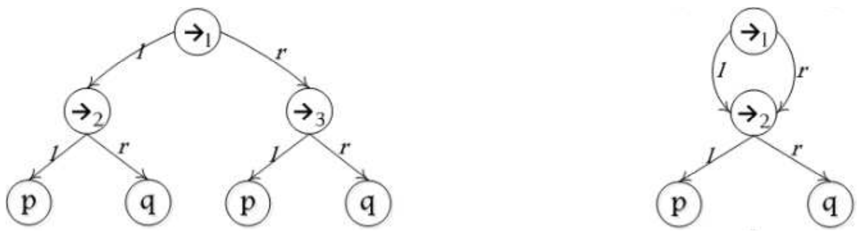


Figure 3.1: Formula $(p \to q) \to (p \to q)$ depicted as a formula tree (left side) and as a formula graph (right side).

The formula nodes (F-nodes) are labeled with formulas as being encoded/represented by their principal connectives (in particular, atoms). Additionally we will use delimiter nodes (D-nodes) H_i and C to indicate which are the hypothesis formulas and the final conclusion formula of the mimp-graph.

As for the inference-part of a mimp-graph we have the rule nodes (R-nodes) that are labeled with the names of the inference rules (→I and →E). Both logic connectives and inference names may be indexed, in order to achieve an 1–1 correspondence between formulas (inferences) and their representations (names).

Since formulas are uniquely determined by the representations in question, i.e. F-node labels, in the sequel we will sometimes identify both; to emphasize the difference we will refer to the former as *formula graphs*, i.e. those whose F-node labels are formulas, instead of principal connectives.

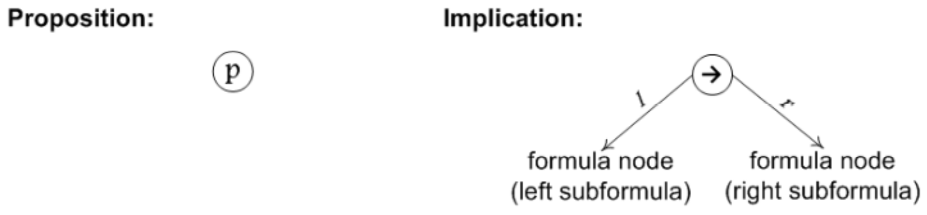


Figure 3.2: Types of formula nodes.

Edges are labeled with tokens that identify the connections between the respective R-nodes and F-nodes. Note that formulas may occur only once in the mimp-graph. Subformulas are indicated by outgoing edges with labels l (left) and r (right), see Figure 3.2.

R-nodes, like in Natural Deduction, require the correct number of premises. The premises are indicated by incoming edges and there are edges from the R-nodes to the conclusion formulas. In the terminology about R-nodes, the R-node →E has two incoming edges, these are distinguished by calling them major (with label M) or minor (with label m) and so also the F-node 'premise' associated with these edges. Thus, an R-node →E has a major premise and a minor premise, the major premise contains the connective that is eliminated; the other premise is called 'minor', and an R-node →I has only one premise. Figure 3.3 shows the R-nodes →I (implication introduction), →Iv[1] (implication introduction vacuously) and →E (implication elimination). In the case in which the discharge of hypotheses is vacuous, a mimp-graph is represented by a disconnected graph, where the discharged F-node is not linked to the conclusion of the rule by any directed path.

In the R-nodes, formulas are re-used, which is indicated by putting several arrows towards them, hence the number of edges with label p, M, m and c coming/going to an F-node could be arbitrarily large. To make all this a bit

[1]the "v" stands for "vacuous", this case of the rule →I discharges a hypothesis vacuously.

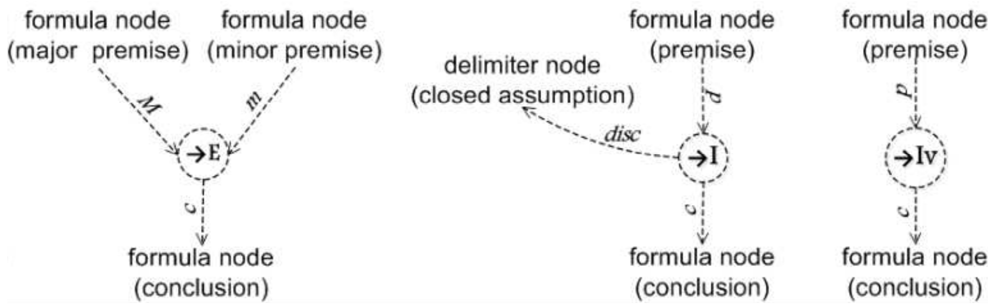


Figure 3.3: Types of rule nodes of the mimp-graph.

$$\dfrac{\dfrac{\dfrac{\dfrac{[p]^1 \quad p \to q}{q}\to E_1 \quad [q \to r]^2}{r}\to E_2}{p \to r}(\to I_3,1)}{(q \to r) \to (p \to r)}(\to I_4,2)$$

$$\Downarrow \textit{trans}$$

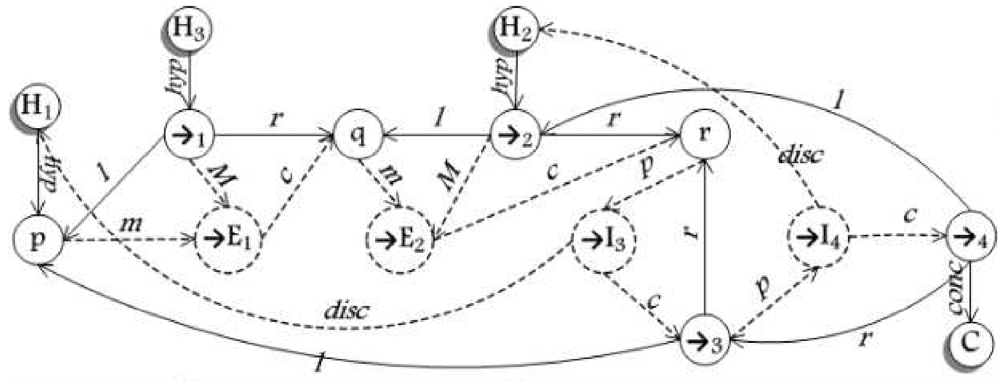


Figure 3.4: The transition from a derivation in ND to a mimp-graph.

more intuitive we give an example of a mimp-graph in Figure 3.4, which can be seen as a derivation of $(q \to r) \to (p \to r)$ from $(p \to q)$. Hypotheses are replaced by D-nodes H and indexes of discarded hypotheses are replaced by additional edges assigned with the label: *disc*. Note that the D-node H can only be discharged once. The re-using of formulas is necessary. We remind the reader that some valid implicational formulas, such as $((((r \to s) \to r) \to r) \to s) \to s$ (see Figure 3.5), need to use any number of times a subformula, in this case the subformula $(((r \to s) \to r) \to r) \to s$ is used twice. Because of this, edges p, m, M and c in Figure 3.5 are indexed with the same index of the R-node to which they belong.

F-nodes in the graph (Figure 3.4) are labeled with propositional letters p, q and r, the connective $\to$; R-nodes are labeled with $\to$E and $\to$I. The underlying idea is that there is an *inferential order* between R-nodes that provides the corresponding derivability order; F-node labeled $\to_4$ linked to the delimiter node

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[r]^1}{((r\to s)\to r)\to r} \quad [(((r\to s)\to r)\to r)\to s]^3}{s}}{r\to s} \qquad [(r\to s)\to r]^2}{r}}{(((r\to s)\to r)\to r)} \qquad [(((r\to s)\to r)\to r)\to s]^3}{s}}{((((r\to s)\to r)\to r)\to s)\to s}$$

$\Downarrow$ *trans*

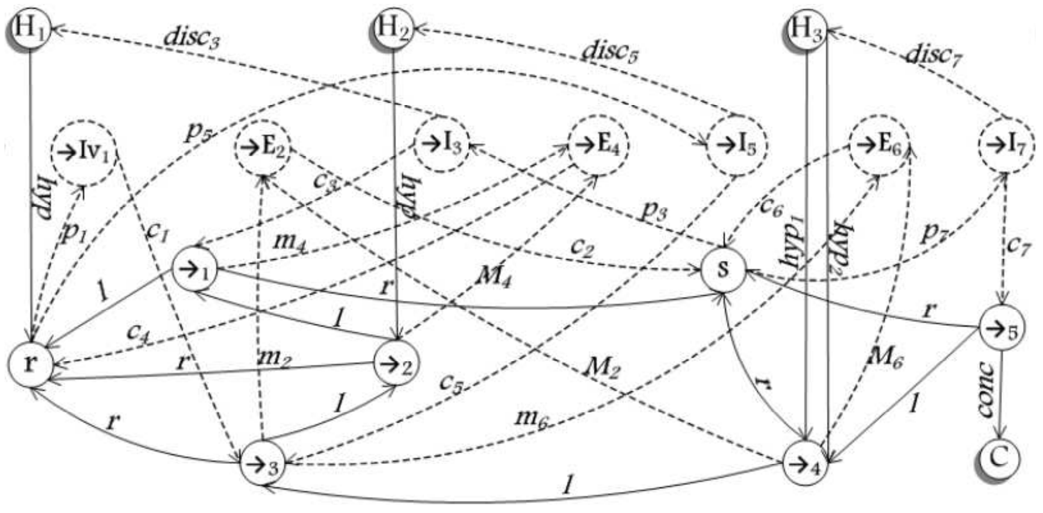


Figure 3.5: The transition from a natural deduction proof to a mimp-graph.

with labeled C by an edge labeled *conc* is the root node and the conclusion of the proof represented by the graph. Besides, the node $\to_1$ linked to the delimiter node labeled H_3 by the edge labeled *hyp* (hypothesis) in the graph is representing the premise $(p \to q)$.

We want to emphasize that the mimp-graphs put together information on formula-graphs and R-nodes. To make it more transparent we can use different types of lines. In this way F-nodes and edges between them are used solid lines, whereas inference nodes and edges between them and adjacent premises and/or conclusions are used dashed lines and additionally delimiter nodes have been shaded. So nodes of types $\to$ and p (propositions) together with adjacent edges (l, r) have solid lines, whereas nodes labeled $\to$I and $\to$E together with adjacent edges $(m, M, p, c, disc)$ have dashed lines.

We now give a formal definition of mimp-graphs.

Definition 5 (Label types) *There are four types of labels:*

- R-Labels *is the set of inference labels:* $\{\to\mathrm{I}_n/n \in \mathbb{N}\} \cup \{\to\mathrm{E}_m/m \in \mathbb{N}\}$,
- F-Labels *is the set of formula labels:* $\{\to_i/i \in \mathbb{N}\}$ *and propositional letters* $\{p, q, r, ...\}$,

- E_F-Labels *is the set of edge labels:* $\{l$ *(left),* r *(right)*$\}$,
- E_M-Labels *is the set of edge labels:* $\{$ p_i *(premise),* m_j *(minor premise),* M_k *(major premise),* c_r *(conclusion),* $disc_s$ *(discharge),* hyp_t *(hypothesis), conc (final conclusion)* $/\, i, j, k, r, s, t \in \mathbb{N}\}$,
- D-Labels *is the set of delimiter labels:* $\{H_k / k \in \mathbb{N}\} \cup \{C\}$.

The union of these four sets of label types will be called LBL.

Definition 6 *Let G be a graph.* l_V *is a labeling function from the* nodes *of G to* R∪F∪D-Labels, *i.e. it assigns a label to each node;* l_E *is a labeling function from the* edges *of G to* $\mathrm{E}_F \cup \mathrm{E}_M$-Labels, *i.e. it assigns a label to each edge.*

Definition 7 *Let* $G_1 = \langle \mathrm{V}^1, \mathrm{E}^1, \mathrm{L}^1 \rangle$ *and* $G_2 = \langle \mathrm{V}^2, \mathrm{E}^2, \mathrm{L}^2 \rangle$ *be two graphs, where:* V^1 *and* V^2 *are sets of vertices,* E^1 *and* E^2 *are sets of labeled* edges, L^1 *and* L^2 *are subsets of* LBL. *The operation* $G_1 \oplus G_2 = \langle\, \mathrm{V}^1 \sqcup \mathrm{V}^2, \mathrm{E}^1 \not\sqcup \mathrm{E}^2, \mathrm{L}^1 \cup \mathrm{L}^2 \rangle$ *equalizes nodes of* G_1 *with nodes of* G_2 *that have the same label, and equalizes edges with the same source, target and label into one. To be precise, the sets* $\mathrm{V}^1 \sqcup \mathrm{V}^2$ *and* $\mathrm{E}^1 \not\sqcup \mathrm{E}^2$ *are of the form*

- $\mathrm{V}^1 \sqcup \mathrm{V}^2 = \{x_1 \in V^1\} \cup \{x_2 \in V^2 / \forall x_1 \in V^1\ l_{V^1}(x_1) \neq l_{V^2}(x_2)\}$.
- $\mathrm{E}^1 \not\sqcup \mathrm{E}^2 = \{x_1 \xrightarrow{t_1} y_1 \in E^1\} \cup \{x_2 \xrightarrow{t_2} y_2 \in E^2 /\ \forall (x_1 \xrightarrow{t_1} y_1) \in E^1\ (l_{V^1}(x_1) \neq l_{V^2}(x_2) \vee l_{E^1}(x_1 \xrightarrow{t_1} y_1) \neq l_{E^2}(x_2 \xrightarrow{t_2} y_2) \vee l_{V^1}(y_1) \neq l_{V^2}(y_2))\}$.

Note: This operation does not include nodes created by duplication or marked, according what is explained in Definition 28.

We will use the terms α_m, β_n and γ_r to represent the principal connective of the formula α, β and γ respectively. In next definitions, the graph has been simplified to improve readability, and to explain the details.

Definition 8 (Formula graph) *A* formula graph *G is a directed graph* $\langle$N, A, B$\rangle$ *where:* N *is a set of vertices (or nodes),* A *is a set of labeled* edges $\langle v \in \mathrm{N},\ t \in \mathrm{E}_F\text{-Labels},\ v' \in N \rangle$ *of source v, target v' and label t and is identified with the arrow* $v \xrightarrow{t} v'$, B *is a set of labels* $b \in \mathrm{F} \cup \mathrm{E}_F$-Labels.

Formula graphs are recursively defined as follows:

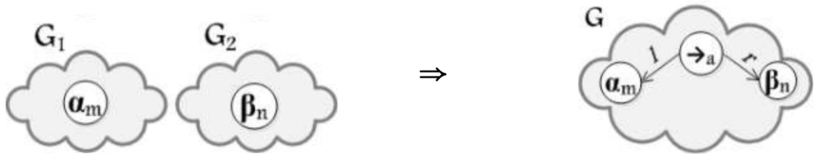
Basis *One propositional letter* p *is a formula graph.*

➔ *If* G_1 *is a formula graph with root node* α_m *and* G_2 *is a formula graph with root node* β_n*, then the graph G that is defined as* $G_1 \oplus G_2$ *with*

1. *an F-node* $\rightarrow_q$,

2. edges: $\rightarrow_q \xrightarrow{l} \alpha_m$ *and* $\rightarrow_q \xrightarrow{r} \beta_n$*,*

is a formula graph (see the following figure).



Definition 9 (Mimp-graph) *A* mimp-graph *G is a directed graph* ⟨V, E, L⟩ *where:* V *is a set of nodes,* L *is a subset of* LBL*,* E *is a set of labeled* edges ⟨*v* ∈ V*, t* ∈ $\mathrm{E}_{F\cup M}$-Labels*, v′* ∈ V ⟩ *of source v, target v′ and label t and identified with the arrow* $v \xrightarrow{t} v'$*.*

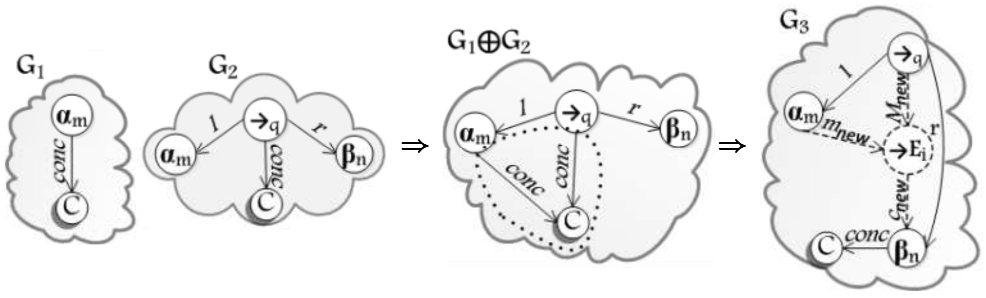
Mimp-graphs are recursively defined as follows:

Basis *If* G_1 *is a formula graph with root node* α_m *then the graph* G_2 *defined as* G_1 *with delimiter nodes* H_n *and* C *and edges* $\alpha_m \xrightarrow{conc} C$ *and* $H_n \xrightarrow{hyp} \alpha_m$ *is a mimp-graph.*

→E *If* G_1 *and* G_2 *are mimp-graphs, and the graph (intermediate step) obtained by* $G_1 \oplus G_2$ *contains the edge* $\rightarrow_q \xrightarrow{l} \alpha_m$ *and two nodes* $\rightarrow_q$ *and* α_m *linked to the delimiter node* C*, then the graph* G_3 *that is defined as* $G_1 \oplus G_2$ *with*

1. *the removal of ingoing edges in the node* C *which were generated in the intermediate step (see the figure below, dotted area in* $G_1 \oplus G_2$*);*
2. *an R-node* $\rightarrow\mathrm{E}_i$ *at the top position;*
3. *edges:* $\alpha_m \xrightarrow{m_{new}} \rightarrow\mathrm{E}_i$*,* $\rightarrow_q \xrightarrow{M_{new}} \rightarrow\mathrm{E}_i$*,* $\rightarrow\mathrm{E}_i \xrightarrow{c_{new}} \beta_n$ *and* $\beta_n \xrightarrow{conc} C$*, where new is a fresh (new) index ranging over all edges of kind c, m and M ingoing and/or outgoing of F-nodes* α_m*,* β_n *and* $\rightarrow_q$*;*

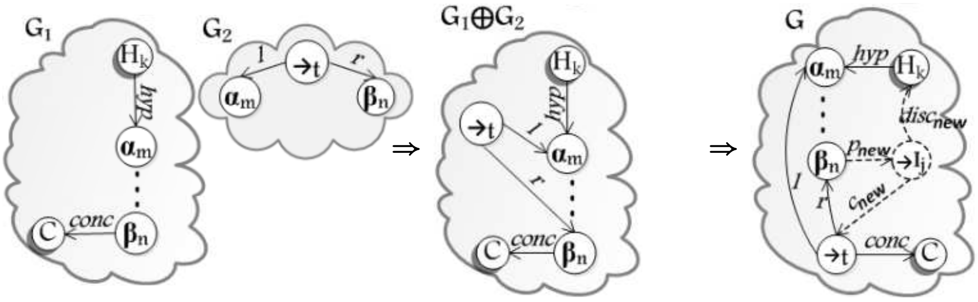
is a mimp-graph (see the following figure).



→I *If* G_1 *is a mimp-graph and contains a node* β_n *linked to the delimiter node* C *and the node* α_m *linked to the delimiter node* H_k*, then the graph* G *that is defined as*

1. *$G := G_1 \oplus G_2$, such that G_2 is a formula graph with root node $\rightarrow_t$ linked to F-nodes α_m and β_n by edges: $\rightarrow_t \xrightarrow{l} \alpha_m$, $\rightarrow_t \xrightarrow{r} \beta_n$;*
2. *with the removal of edges: $\beta_n \xrightarrow{conc} C$;*
3. *an R-node $\rightarrow\mathrm{I}_j$ at the top position;*
4. *edges: $\beta_n \xrightarrow{p_{new}} \rightarrow\mathrm{I}_j$, $\rightarrow\mathrm{I}_j \xrightarrow{c_{new}} \rightarrow_t$, $\rightarrow_t \xrightarrow{conc} C$ and $\rightarrow\mathrm{I}_j \xrightarrow{disc_{new}} H_k$, where new is a fresh (new) index considering all edges of kind p, disc and c ingoing and/or outgoing of F-nodes α_m, β_n and $\rightarrow_q$;*

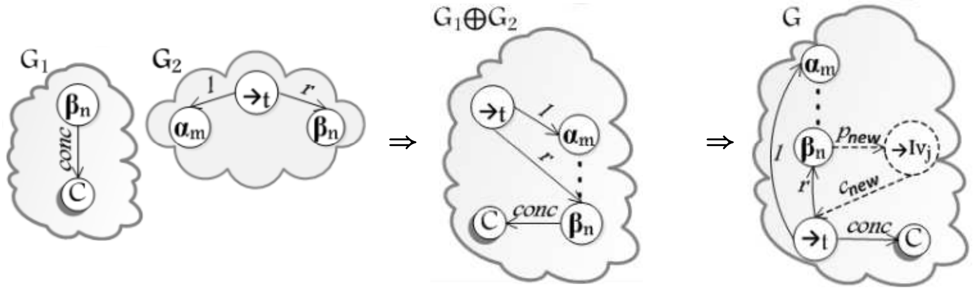
is a mimp-graph (see the following figure; the α_m-node is discharged*).*



$\rightarrow$Iv *If G_1 is a mimp-graph and contains a node β_n linked to the delimiter node C, then the graph G that is defined as*

1. *$G := G_1 \oplus G_2$, such that G_2 is a formula graph with root node $\rightarrow_t$ linked to F-nodes α_m and β_n by edges: $\rightarrow_t \xrightarrow{l} \alpha_m$, $\rightarrow_t \xrightarrow{r} \beta_n$;*
2. *with the removal of the edge $\beta_n \xrightarrow{conc} C$;*
3. *an R-node $\rightarrow\mathrm{Iv}_j$ at the top position;*
4. *edges: $\beta_n \xrightarrow{p_{new}} \rightarrow\mathrm{Iv}_j$, $\rightarrow_t \xrightarrow{conc} C$ and $\rightarrow\mathrm{Iv}_j \xrightarrow{c_{new}} \rightarrow_t$, where new is an index under the same conditions of the previous case;*

is a mimp-graph (see the following figure).



Definition 10 *Let G be a mimp-graph. An* inferential order $<$ *on nodes of G is a partial ordering of the R-nodes of G such that $n < n'$ iff n and n' are*

R-nodes and there is an F-node f such that $n \xrightarrow{lbl_1} f \xrightarrow{lbl_2} n'$ and lbl_1 is c and lbl_2 is m, or lbl_1 is c and lbl_2 is M, or lbl_1 is c and lbl_2 is p. Node n is a top position *node if n is maximal w.r.t. $<$.*

In order to avoid overloading of indexes, we will omit, whenever possible, the indexing of edges of kind c, m, M, p and $disc$, keeping in mind that the coherence of indexing is established by the kind of rule-node to which they are linked.

Lemma 1 enables us to prove that a given graph G is a mimp-graph. We just have to check that G has an inferential ordering on all R-nodes and that each node of G is of one of the possible types that generate the Basis and the construction cases $\rightarrow$E, $\rightarrow$I and $\rightarrow$Iv of Definition 9.

Lemma 1 *G is a mimp-graph if and only if the following properties hold:*

1. *There exists a well-founded (hence acyclic) inferential order $<$ on all R-nodes of the mimp-graph*[2]*.*

2. *Every node N of G is of one of the following six types:*

   P *N is labeled with one of the propositional letters: $\{p,\ q,\ r,\ \dots\ \}$. N has no outgoing edges l and r.*

   K *N has label $\rightarrow_n$ and has exactly two outgoing edges with label l and r, respectively. N may has outgoing edges with labels p_i, m_j or M_k; and ingoing edges with label c_l and hyp_m.*

   E *N has label $\rightarrow\mathrm{E}_i$ and has exactly one outgoing edge $\rightarrow\mathrm{E}_i \xrightarrow{c} \beta_n$, where β_n is a node type* **P** *or* **K***. N has exactly two ingoing edges $\alpha_m \xrightarrow{m} \rightarrow\mathrm{E}_i$ and $\rightarrow_q \xrightarrow{M} \rightarrow\mathrm{E}_i$, where α_m is a node type* **P** *or* **K***. There are two outgoing edges from the node $\rightarrow_q$: $\rightarrow_q \xrightarrow{l} \alpha_m$ and $\rightarrow_q \xrightarrow{r} \beta_n$.*

   I *N has label $\rightarrow\mathrm{I}_j$ (or $\rightarrow\mathrm{Iv}_j$, if discharges an hypothesis vacuously), has one outgoing edge $\rightarrow\mathrm{I}_j \xrightarrow{c} \rightarrow_t$, and one (or zero for the case $\rightarrow$Iv) outgoing edge $\rightarrow\mathrm{I}_j \xrightarrow{disc} H_k$. N has exactly one ingoing edge: $\beta_n \xrightarrow{p} \mathrm{I}_j$, where β_n is a node type* **P** *or* **K***. There are two outgoing edges from the node $\rightarrow_t$: $\rightarrow_t \xrightarrow{l} \alpha_m$ and $\rightarrow_t \xrightarrow{r} \beta_n$ such that there is one (or zero for the case $\rightarrow$Iv) ingoing edge to the node α_m: $\mathrm{H}_k \xrightarrow{hyp} \alpha_m$.*

   H *N has label H_k and has outgoing edges with label hyp.*

[2]We can extend this "dashed" inferential order $<$ to the full "mixed" order $<^*$ by adding new "solid" relations $<$ corresponding to arrows $\xrightarrow{l}$ and $\xrightarrow{r}$ between F-nodes. Note that $<^*$ may contain cycles (see Figure 3.4). However all recursive definitions and inductive proofs to follow are based on the well-founded "dashed" order $<$, hence being legitimate.

C *N has label* C *and has exactly one ingoing edge with label conc.*

Proof:

$\Rightarrow$: By induction on the construction of mimp-graph (Definition 9). For every construction case for mimp-graphs we have to check the three properties stated in Lemma. Property (2) is immediate. For property (1), we know from the induction hypothesis that there is an inferential order $<$ on R-nodes of the mimp-graph. In the construction cases $\rightarrow$I, $\rightarrow$Iv or $\rightarrow$E, we make the new R-node that is introduced highest in the $<$-ordering, which yields an inferential ordering on R-nodes. In the construction case $\rightarrow$E, when we have two inferential orderings, $<_1$ on G_1 and $<_2$ on G_2. Then $G_1 \oplus G_2$ can be given an inferential ordering by taking the union of $<_1$ and $<_2$ and in addition putting $n < m$ for every R-node n, m such that $n \in G_1$ and $m \in G_2$.

$\Leftarrow$: By induction on the number of R-nodes of G. Let $<$ be the topological order that is assumed to exist. Let n be the R-node that is maximal w.r.t. $<$. Then n must be on the top position. When we remove node n, including its edges linked (if n is of type **I**) and the node type **C** is linked to the premise of the R-node, we obtain a graph G' that satisfies the properties listed in Lemma. By induction hypothesis we see that G' is a mimp-graph. Now we can add the node n again, using one of the construction cases for mimp-graphs: *Basis* if n is a **P** node or **K** node, $\rightarrow$E if n is an **E** node, $\rightarrow$I if n is an **I** node. ■

It is natural to consider minimal mimp-graph-like representations of given natural deductions. Actually one can try to minimize the number of F-nodes and/or R-nodes, but in this version we consider only the F-option, as it helps to reduce the size under standard normalization (see the next section). To grasp the point, note that mimp-graph in Figure 3.4 (see above) is F-minimal, i.e., its F-labeled nodes refer to pairwise distinct formulas. This observation is summarized by Theorem 3.

Theorem 3 (F-minimal representation) *Every standard tree-like natural deduction* Π *has a uniquely determined (up to graph-isomorphism)* F-minimal mimp-like representation G_Π *that satisfies the following four conditions.*

1. G_Π *is a mimp-graph whose size does not exceed the size of* Π.
2. Π *and* G_Π *both have the same (set of) hypotheses and the same conclusion.*
3. *There is graph homomorphism* $h : \Pi \rightarrow G_\Pi$ *that is injective on R-Labels.*
4. *All F-Labels occurring in* G_Π *denote pairwise distinct formulas.*

Proof: Let N and Form be the set of nodes and formulas, respectively, occurring in Π. Note that Π determines a fixed surjection $f : \mathrm{N} \rightarrow \mathrm{Form}$ that may not be injective (for in Π, one and the same formula may be assigned to different nodes). In order to obtain G_Π take as R-nodes the inferences occurring in Π assigned with the corresponding R-Labels representing inferences' names (possibly indexed, in order to achieve a 1–1 correspondence between inferences and R-Labels, cf. Figure 3.4). Define basic F-nodes of G_Π as formulas from Form assigned with the corresponding F-Labels representing formulas' principal connectives (possibly indexed, in order to achieve an 1–1 correspondence between formulas and F-Labels, cf. Figure 3.4). So the total number of all basic F-nodes of G_Π is the cardinality of the set Form, while f being a mapping from nodes of Π onto the basic F-nodes of G_Π. To complete the construction of G_Π we add, if necessary, the remaining F-nodes labeled by failing representations of subformulas of $f(x)$, $x \in \mathrm{N}$, and define the E-Labels of G_Π, accordingly. Note that by the definition all nodes of G_Π have pairwise distinct labels. In particular, every F-Label occurs only once in G_Π, which yields the crucial condition 4. ■

3.2 Normalization for mimp-graphs

In this section we define the normalization procedure for mimp-graphs. It is based on the standard normalization method given by Prawitz. Thus a *maximal formula* in mimp-graphs is a $\rightarrow$I followed by a $\rightarrow$E of the same formula graph (see Definition 11). It is the same notion of maximal formulas that is being used in natural deduction derivations. So a maximal formula occurrence is the consequence of an application of an introduction rule and major premise of an application of an elimination rule. But here we assume that derivations represented by mimp-graphs. We wish to eliminate such maximal formula by dropping nodes and edges that are involved in the maximal formula.

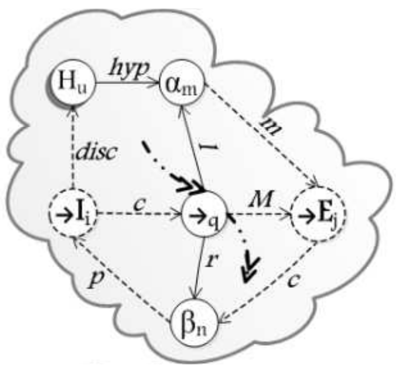
Definition 11 *A* maximal formula *m in a mimp-graph G (see the figure below, where the double-headed arrows represent several edges) is a sub-graph of G consisting of:*

1. *F-nodes α_m, β_n, $\rightarrow_q$, the R-node $\rightarrow\mathrm{I}_i$ and the delimiter node H_u;*

2. *the R-node $\rightarrow\mathrm{E}_j$ at the top position;*

3. *edges: $\rightarrow_q \xrightarrow{l} \alpha_m$, $\rightarrow_q \xrightarrow{r} \beta_n$, $\beta_n \xrightarrow{p} \rightarrow\mathrm{I}_i$, $\rightarrow\mathrm{I}_i \xrightarrow{c} \rightarrow_q$, $H_u \xrightarrow{hyp} \alpha_m$, $\rightarrow\mathrm{I}_i \xrightarrow{disc} H_u$, $\alpha_m \xrightarrow{m} \rightarrow\mathrm{E}_j$, $\rightarrow_q \xrightarrow{M} \rightarrow\mathrm{E}_j$ and $\rightarrow\mathrm{E}_j \xrightarrow{c} \beta_n$;*

$$\dfrac{\begin{matrix}\Pi_1\\ \alpha\end{matrix} \qquad \dfrac{\begin{matrix}[\alpha]^u\\ \Pi_2\\ \beta\end{matrix}}{\alpha \to \beta}\,(\to\text{I, u})}{\begin{matrix}\beta\\ \Pi_3\end{matrix}}\,\to\text{E} \qquad \Rightarrow$$



However, as a special case of maximal formula could also happen that between the R-nodes →I and →E there are several other maximal formulas such as the case in the example of Figure 3.6, where there is a maximal formula with R-nodes $\to I_4$ and $\to E_5$ (dotted area with white background) inside of the maximal formula with R-nodes $\to I_3$ and $\to E_6$ (area with shaded background). That is, the inferential orders of R-nodes are intermediate to those of the R-nodes $\to I_3$ and $\to E_6$. In these cases eliminate in one step the maximal formula, with the exception of the R-nodes $\to_1$ and $\to_2$ because they are still related with other nodes, becoming as shown in the same figure. We can visualize another maximal formula (with R-nodes $\to I_9$ and $\to E_{10}$) that is pending removal.

Definition 12 *(1) For $n_i \in V$, a* path *in a proof-graph is a sequence of vertices and edges of the form $n_1 \xrightarrow{l_1} n_2 \xrightarrow{l_2} ... \xrightarrow{l_{k-2}} n_{k-1} \xrightarrow{l_{k-1}} n_k$ such that n_1 is a hypothesis F-node, n_k is the conclusion F-node, n_i alternating between an R-node and an F-node. edges l_i alternate between two types of edges: $l_j \in \{m, M, p\}$ and $l_j = c$. (2) A* branch *is an initial part of a* path *which stops at the conclusion F-node or at the first minor premise whose major premise is the conclusion of an R-node.*

The term *R-node sequence* is representing a deduction, and if it is a smaller part of another R-node sequence (subdeduction), then it is called *a subsequence* of the latter. A subsequence that derives a premise of the last R-node application in an R-node sequence is called a direct R-node subsequence. Instead of writing "the direct R-node subsequence that derives the minor premise of the last inference of an R-node sequence D", we simply write *"the minor subsequence of D".*

Definition 13 *A reordering of a given mimp-graph G is obtaining by supplying G with the following (new) inferencial order on the R-nodes of G.*

- $o(t_m) = 0$ *for an R-node t_m starting with hypothesis.*
- $o(t) = o(t') + 1$ *if the conclusion formula of R-node t' is premise, right premise or major premise of t.*

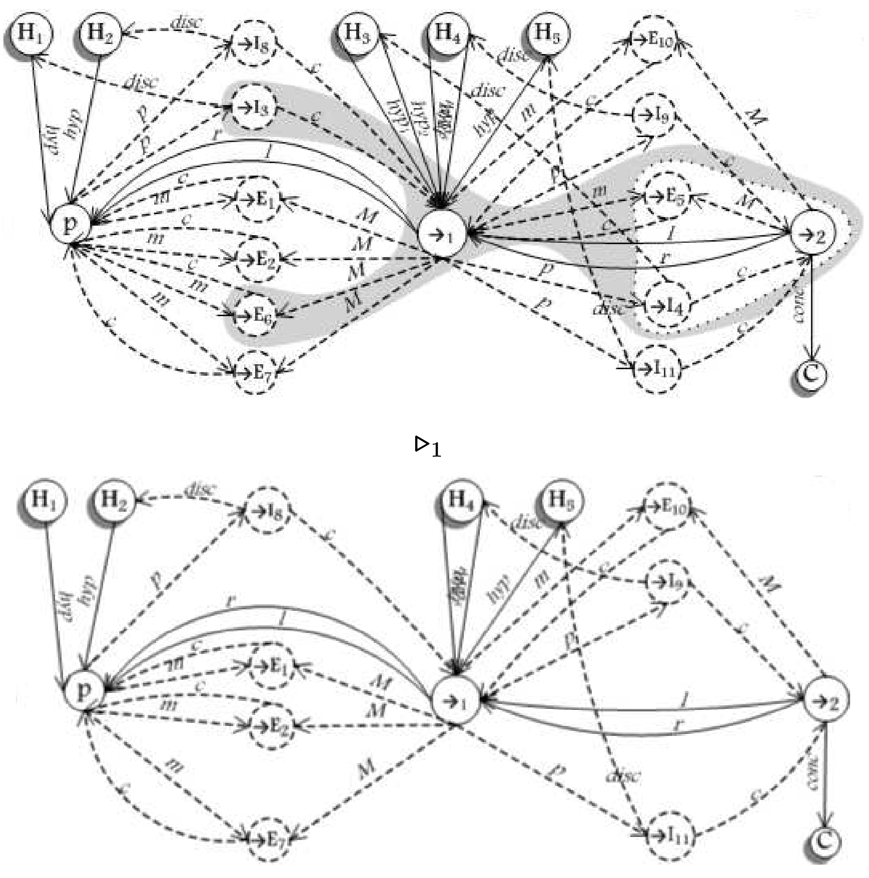


Figure 3.6: Example of a maximal formula with an intermediate maximal formula.

Proposition 1 *A graph obtained by a reordering according to Definition 13 is a mimp-graph.*

Definition 14 *Consider a mimp-graph G with a maximal formula m, that satisfies the following requirements:*

1. *Between the R-nodes $\rightarrow\mathrm{I}_i$ and $\rightarrow\mathrm{E}_l$ there are zero or more maximal formulas with inferential orders within the range of these rule nodes.*
2. *There is an edge $\rightarrow\mathrm{I}_i \xrightarrow{c} \rightarrow_q$ and the F-node $\rightarrow_q$ has zero or more ingoing edges.*
3. *There is an edge $\rightarrow_q \xrightarrow{M} \rightarrow\mathrm{E}_l$ and the F-node $\rightarrow_q$ is the premise of zero or more of another R-nodes.*
4. *If a branch will be separated from the inferential order this branch must be insertable in the following branch, according to the order, i.e., the conclusion of this separated branch is the premise in the following branch.*

The elimination of a maximal formula *m from G is the following operation on a mimp-graph (see Figure 3.7, the double-headed arrows are representing several edges):*

1. *If there is no maximal formula between the R-nodes* $\rightarrow\mathrm{I}_i$ *and* $\rightarrow\mathrm{E}_l$ *then follow these steps:*

   (a) *If the edge* $\rightarrow\mathrm{I}_i \xrightarrow{c} \rightarrow_q$ *is the only ingoing edge to* $\rightarrow_q$ *and the edge* $\rightarrow_q \xrightarrow{M} \rightarrow\mathrm{E}_l$ *is the only outgoing edge from* $\rightarrow_q$ *then remove edges to and from the F-node* $\rightarrow_q$*, and remove the F-node* $\rightarrow_q$*.*

   (b) *If the D-node* H_u*, discharged by* $\rightarrow\mathrm{I}$*, has n outgoing edges with label hyp then repeat n-times edges in the minor subsequence of* $\rightarrow\mathrm{E}_l$*.*

   (c) *Remove edges to and from nodes* $\rightarrow\mathrm{I}_i$*,* $\rightarrow\mathrm{E}_l$ *and* H_u*.*

   (d) *Remove nodes* $\rightarrow\mathrm{I}_i$*,* $\rightarrow\mathrm{E}_l$ *and* H_u*.*

   (e) *Apply the operation defined in Definition 13 to the resulting graph. Note that Proposition 1 ensures that the result is a mimp-graph.*

2. *Otherwise eliminate the maximal formulas between the R-nodes* $\rightarrow\mathrm{I}_i$ *and* $\rightarrow\mathrm{E}_l$ *as in the previous step.*

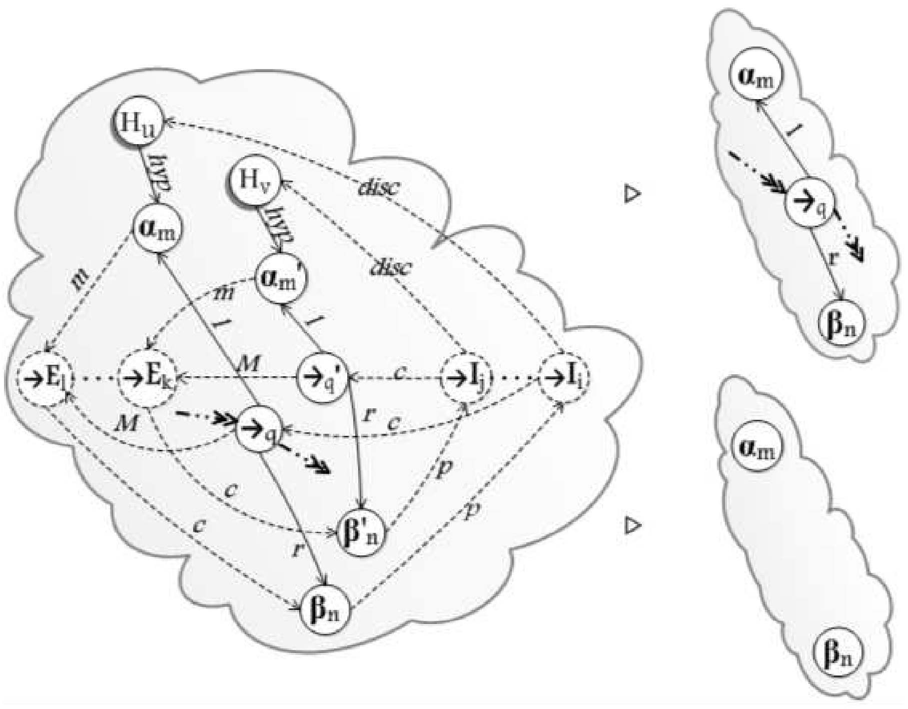


Figure 3.7: Elimination of a maximal formula in mimp-graphs.

Note that the removal of a node type I generated by case →Iv in Definition 23, disconnects the graph, meaning that the sub-graph hypotheses linked, by the edge with label m, to the node labeled →E removed, is no longer connected to the delimiter node type C.

Figure 3.8 shows an instance of the elimination of a maximal formula in tree form. Note that this example shows the reason why essentially our (weak)

$$\cfrac{\begin{matrix}\Pi_2\\ \beta\end{matrix} \qquad \cfrac{\begin{matrix}\Pi_1\\ \alpha\end{matrix} \qquad \cfrac{\cfrac{\begin{matrix}[\beta]^v[\alpha]^u\\ \Pi_0\\ \gamma\end{matrix}}{\beta \to \gamma}\,v}{\alpha \to (\beta \to \gamma)}\,u}{\beta \to \gamma}}{\gamma} \quad \rhd_1 \quad \cfrac{\begin{matrix}\Pi_2\\ \beta\end{matrix} \qquad \cfrac{\begin{matrix}[\beta]^v & \begin{matrix}\Pi_1\\ \alpha\end{matrix}\\ \multicolumn{2}{c}{\Pi_0}\\ \multicolumn{2}{c}{\gamma}\end{matrix}}{\beta \to \gamma}}{\gamma} \quad \rhd_2 \quad \begin{matrix}\Pi_2 & \Pi_1\\ \beta & \alpha\\ \multicolumn{2}{c}{\Pi_0}\\ \multicolumn{2}{c}{\gamma}\end{matrix}$$

$$\Downarrow trans$$

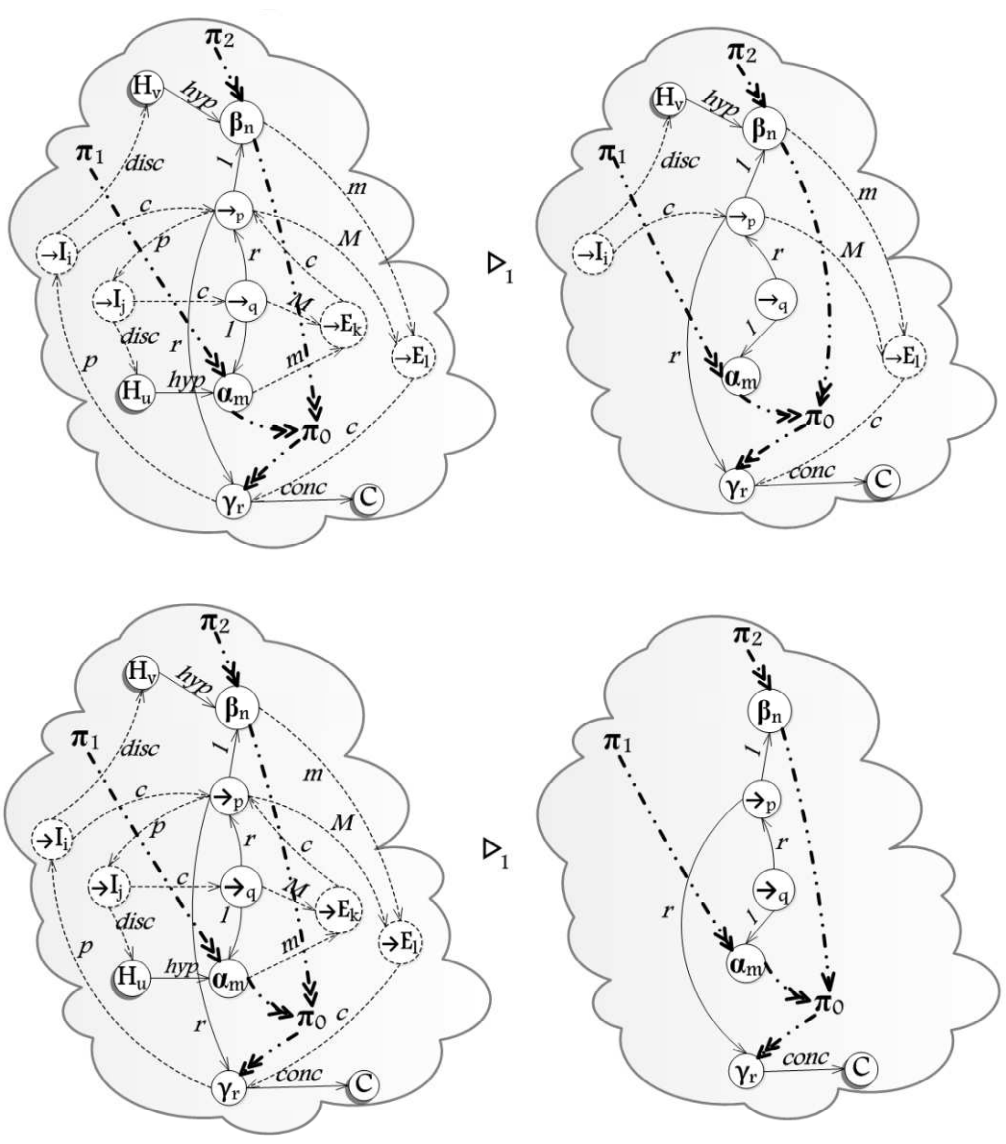


Figure 3.8: Eliminating a maximal formula in a natural deduction proof and its mimp-graph translation.

normalization theorem is directly a strong normalization theorem. The formula $\beta \rightarrow \gamma$ is not a maximal formula before a reduction is applied to eliminate the maximal formula $\alpha \rightarrow (\beta \rightarrow \gamma)$. This possibility of having hidden maximal formulas in Natural Deduction is the main reason to use more sophisticated methods whenever proving strong normalization. In mimp-graphs there is no possibility to hide a maximal formula because all formulas are represented only once in the graph. In this graph $\beta \rightarrow \gamma$ is already a maximal formula. We can choose to remove any of the two maximal formulas. If $\beta \rightarrow \gamma$ is chosen to be eliminated, by the mimp-graph normalization procedure, its reduction eliminates the $\alpha \rightarrow (\beta \rightarrow \gamma)$ too. On the other hand, the choice of $\alpha \rightarrow (\beta \rightarrow \gamma)$ to be reduced only eliminates itself. In any case, the number of maximal formula decreases.

Lemma 2 *If G is a mimp-graph with a maximal formula m and G' is obtained from G by eliminating m, then G' is also a mimp-graph.*

Proof: We use Lemma 1. All nodes in G' are of the right form: P, F, E, I, H or C. We verify that G' has one ingoing edge with label *conc* to the delimiter node type C and that is acyclic and connected. Finally, a referential order on G' (as defined in Definition 13) between R-nodes must preserve. ■

We shall construct the normalization proof for mimp-graphs. This proof is guided by the normalization measure. That is, the general mechanism from the proof determines that a given mimp-graph G should be transformed into a non-redundant mimp-graph by applying reduction steps and at each reduction step the measure must be decreased. The normalization measure will be the number of maximal formulas in the mimp-graph.

Also note an important observation concerning F-minimal mimp representations (see Theorem 3). Since F-minimal mimps can have at most one occurrence of hypotheses α and/or β, every proper reduction step will diminish the size of deduction. Hence the size of the graph (= the number of nodes) can serve as another inductive parameter, provided that the normalization is being applied to F-minimal mimp-graph representations.

Theorem 4 (Normalization) *Every mimp-graph G can be reduced to a normal mimp-graph G' having the same hypotheses and conclusion as G. Moreover, for any standard tree-like natural deduction Π, if $G := G_\Pi$ (the F-minimal mimp-like representation of Π, cf. Theorem 3), then the size of G' does not exceed the size of G, and hence also Π.*

Remark 1 *The second assertion sharply contrasts to the well-known exponential speed-up of standard normalization. Note that the latter is a consequence* *of the tree-like structure of standard deductions having different occurrences of equal hypotheses formulas, whereas all formulas occurring in F-minimal mimp-like representations are pairwise distinct.*

Proof: This characteristic of preservation of the premises and conclusions of the derivation is proved naturally. Through an inspection of each elimination of maximal formula is observed that the reduction step (see Definition 14) of the mimp-graph does not change the set of premises and conclusions (indicated by the delimiter nodes type H and C) of the derivation that is being reduced.

In addition, the demonstration of this theorem has two primary requirements. First, we guarantee that through the elimination of maximal formulas in the mimp-graph, cannot generate more maximal formulas. The second requirement is to guarantee that during the normalization process, the normalization measure adopted is always reduced.

The first requirement is easily verifiable through an inspection of each case in the elimination of maximal formulas. Thus, it is observed that no case produces more maximal formulas. The second requirement is established through the normalization procedure and demonstrated through an analysis of existing cases in the elimination of maximal formulas in mimp-graphs. To support this statement, it is used the notion of normalization measure, we adopt as measure of complexity (induction parameter) the number of maximal formulas $Nmax(G)$. Besides, as already mentioned, working with F-mimimal mimp-graph representations we can use as optional inductive parameter the ordinary size of mimp-graphs. ■

Normalization Process

We know that a specific mimp-graph G can have one or more maximal formulas represented by $M_1, ..., M_n$. Thus, the normalization procedure is:

1. Choose a maximal formula represented by M_k.
2. Identify the respective number of maximal formulas $Nmax(G)$.
3. Eliminate M_k as defined in Definition 14, creating a new graph G.
4. In this elimination, one of the following three cases may occur:

   a) The maximal formula is removed.

   b) The maximal formula is removed but the F-node is maintained, and, $Nmax(G)$ is decreased;

   c) All maximal formulas are removed.

5. Repeat steps 1 to 4 until the normalization measure $Nmax(G)$ is reduced to 0 and G becomes a normal mimp-graph.

Since the process of eliminating a maximal formula on mimp-graphs always ends in the elimination of at least one maximal formula, and with the decrease in the number of vertices of the graph, we can say that this normalization theorem is directly a strong normalization theorem.

3.3 Summary

In this chapter, we introduced the mimp-graph through the main definitions and examples, mainly devised for extracting proof-theoretic properties from proof system. That is, we have tackled one of our research tracks. Mimp-graphs preserve the ability to represent proofs in Natural Deduction and their minimal formula representation is a key feature of the mimp-graph structure, because as we saw earlier, it is easy to determine an upper bounds on the length of reduction sequences leading to normal proofs. It is the number of maximal formulas. This feature is of crucial importance because we intend to use this method in automated theorem provers.

4
Compactifying

This chapter provide a way to compactify the mimp-graphs, proposed in the previous chapter, while keeping a similar structure in the compacted form. Thus, we intend to minimize the number of F-nodes and R-nodes, thereby F-labeled nodes refer to pairwise distinct formulas and sets of R-labeled nodes refer to pairwise distinct subproofs.

4.1
Compactification Process

Parts of a derivation that have a similar structure can be shared, as shown in Figure 4.1, the boxed formula $p \rightarrow q$ is similar to the boxed formula $p \rightarrow r$ and we can see that they have also similar derivations. (da Costa 2007) sketches as unifying sub-proofs where the similarity is determined by the existence of an unifier, thus given two formulas x and y, there is an object z that fits both formulas (see Algorithm 1).

In mimp-graphs we say that formulas are formula graphs and their similarity is determined by the existence of an isomorphism between these formula graphs (see $\rightarrow_3$ and $\rightarrow_1$ in Figure 4.1). So too, the R-node sequences $\rightarrow\text{I}_2$, $\rightarrow\text{E}_3$, $\rightarrow\text{E}_4$ and $\rightarrow\text{I}_6$, $\rightarrow\text{E}_7$, $\rightarrow\text{E}_8$ have a similar structure because premises and conclusion of one sequence are isomorphic to premises and conclusion of the other sequence, hence they are isomorphic as in the Definition 15. In our proof-graphs, the number of formula nodes (F-nodes) was minimized with the sharing operation $\oplus$ (see Definition 7). Now we want to minimize the number of inferences or R-nodes in the graph for this purpose we extend the mimp-graphs (defined in Chapter 3) and define a representation in graphs, which we call *smimp*.

To make it more transparent we use different types of lines. In this way F-nodes and edges between them use solid lines, whereas inference nodes and edges between them and adjacent premises or conclusions use dashed lines and additionally delimiter nodes have been shaded. So nodes of types $\rightarrow$ and p (propositions) together with adjacent edges (l, r) have solid line, whereas nodes labeled $\rightarrow$I and $\rightarrow$E together with adjacent edges $(m, M, p, c, disc)$ have

$$\dfrac{\dfrac{\dfrac{\dfrac{[p]^1 \quad \dfrac{[p]^1 \quad p\rightarrow(p\rightarrow q)}{p\rightarrow q}\rightarrow E_8}{q}\rightarrow E_7}{\boxed{p\rightarrow q}}\rightarrow I_6 \qquad (p\rightarrow q)\rightarrow p}{p}\rightarrow E_5 \qquad \dfrac{\dfrac{[p]^1 \quad \dfrac{[p]^1 \quad p\rightarrow(p\rightarrow r)}{p\rightarrow r}\rightarrow E_4}{r}\rightarrow E_3}{\boxed{p\rightarrow r}}\rightarrow I_2}{r}\rightarrow E_1$$

$$\Downarrow \mathit{trans}$$

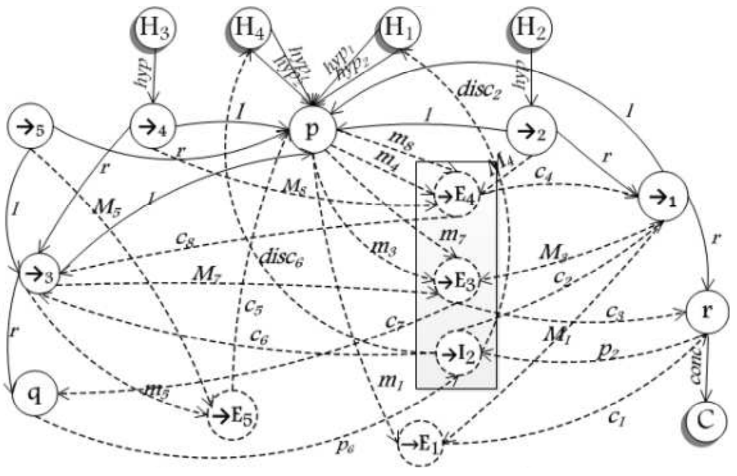


Figure 4.1: The transition from a natural deduction proof to a mimp-graph

dashed lines.

Smimp somehow reflects the sharing of sub-proofs (or derivations with similar conclusion and premises) by means of a graph definition, where the reuse of a sub-result (or sequence of R-nodes) is depicted by a box that contains it. This sharing will be done by comparing of conclusion formula graph with the conclusion of the box that we want to reuse, if they are isomorphic then we proceed to share the box by means of the addition of edges. In the above illustration of Figure 4.2, the F-node $\rightarrow_3$ is the formula that attempts to reuse a derivation with a conclusion isomorphic to it and a premise isomorphic to $\rightarrow_4$, in the below illustration we see how it looks after sharing. Thus, the sequences of rules: $\rightarrow I_2$, $\rightarrow E_3$, $\rightarrow E_4$ and $\rightarrow I_6$, $\rightarrow E_7$, $\rightarrow E_8$, in Figure 4.1, are represented only once as shown in the box, and new ingoing/outgoing edges (type m, M, p, c) of the R-nodes in the box are added with an new index and related with their isomorphic sub-graphs of premises and conclusion that the R-node sequence is sharing.

The graph isomorphism for mimp-graph is a restricted version of the general graph isomorphism that involves deciding the existence of a type of node that preserves the isomorphism between a pair of graphs. For convenience, we add the function $type(v)$ to the definition of mimp-graphs that returns one of the types of nodes described in the Lemma 1.

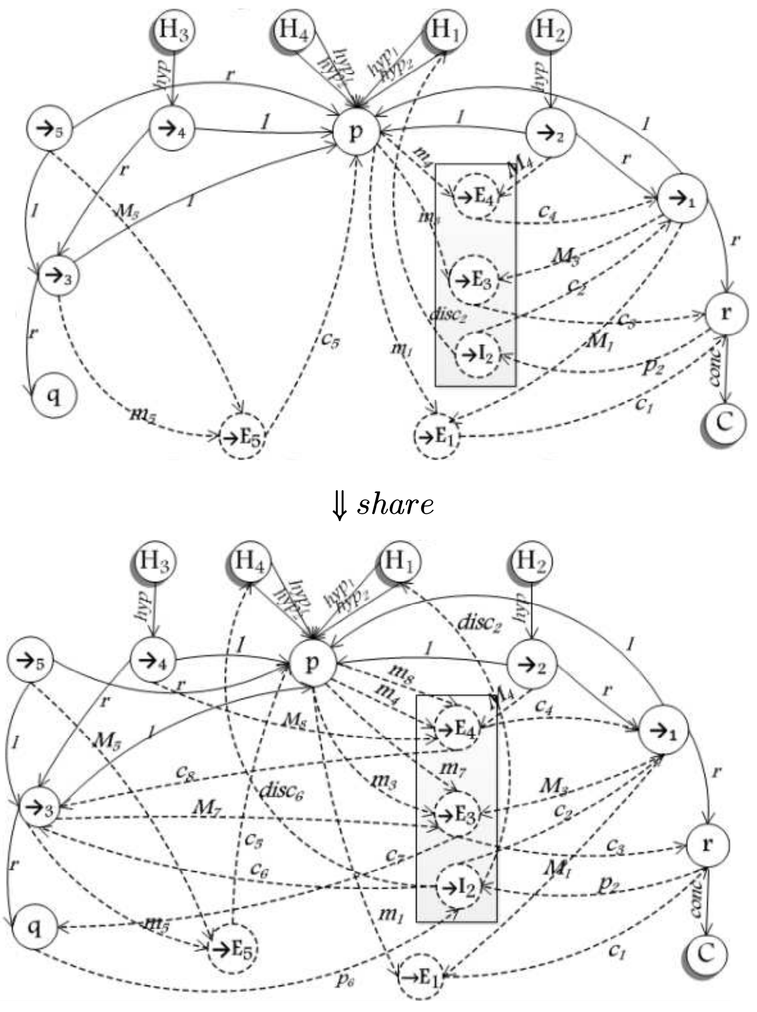


Figure 4.2: The transition before and after of sharing.

Definition 15 (Graph isomorphism) *The isomorphism between a pair of graphs $G = \langle V, E, L\rangle$ and $G' = \langle V', E', L'\rangle$ is a mapping $\phi : V \rightarrow V'$ satisfying the following conditions:*

1. *ϕ is a bijection such that type(v) = type$(\phi(v))$ for all $v \in V$.*
2. *$v_1 \xrightarrow{l} v_2 \in E \leftrightarrow \phi(v_1) \xrightarrow{l'} \phi(v_2) \in E'$ for all $v_1, v_2 \in V'$ such that $l = l'$ unless the index.*

Definition 16 (Subgraph isomorphism) *Given two graphs G_1 and G_2, we say that there is* subgraph isomorphism *from G_1 to G_2 iff there exists a subgraph $S \subset G_2$ such that G_1 and S are isomorphic.*

We present now the known graph transformation: the unfolding. This transformation is to unfold a graph from all its vertices. When a graph contains cycles, this process never stops, theoretically leading to infinite unfoldings. Since formula graphs are acyclic, the unfolding of our graphs is a tree (see right graph of the example in Figure 4.3).

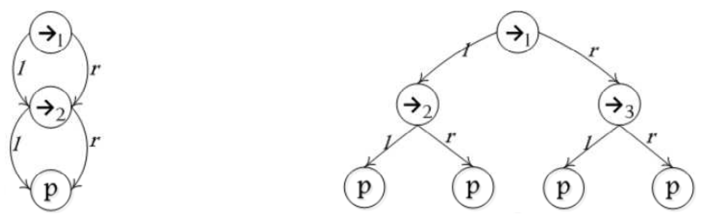


Figure 4.3: Formula $(p \rightarrow p) \rightarrow (p \rightarrow p)$ depicted as a formula graph (left side) and as an unfolding graph (right side).

Definition 17 (Unfolding graph) *The* unfolding graph *of a formula graph is a formula tree that contains the same information, but has no shared nodes. It is obtained by duplicating every node that is the shared target of multiple edges, such that each edge gets its own target node.*

Definition 18 (Substitution for graphs) *A* substitution for graphs σ *is called a unifier for the set of graphs* $\{G_1, ..., G_k\}$*, if and only if* $G_1\sigma = G_2\sigma = ... = G_k\sigma$*. The set* $\{G_1, ..., G_k\}$ *is said unifiable if there is one unifier for it.*

Definition 19 (Pair in Disagreement) *The* pair in disagreement *of a non-empty set of formula graphs S is obtained by locating the nodes (in a pre-order traversal) in the unfolding graph where not all formula graphs in S have exactly the same label in nodes, and then extracting from each formula graph in S the sub-graph with the node occupying this position in disagreement. The set of these respective sub-graphs is the set in disagreement of S.*

Algorithm 1 Unification algorithm adapted by matching

```
1: $k = 0$, $S_k = S$, $D_k = \{\epsilon\}$, $\sigma = \{\epsilon\}$
2: If $S_k$ is a unitary set then substitute the original variables (any remaining)
   of $S$ by new variables applying $\alpha_k/(v_k, v_k)$ for each remaining original
   variable $v_k$ and add $\alpha_k/(v_k, v_k)$ to $\sigma_k$; $\sigma_k$ is the unifier of $S$. Otherwise,
   if $S_k$ is not a unitary set, then find the pair in disagreement $D_k$ of $S_k$.
3: If there are elements $v_k$ and $t_k$ in $D_k$ such that $v_k$ is a variable that does
   not occur in $t_k$, go to step 4. Otherwise, stop, $S$ is not unifiable.
4: If $D_k \notin S_k$, build $S_{k+1}$ by substituting of occurrences of $D_k$ in $S_k$ by $\alpha_k$,
   where $\alpha_k$ is a variable that is neither in $S$ nor in $S_k$. Otherwise, build $S_{k+1}$
   by substituting of occurrences of $D_k$ in $S_k$ by $\alpha_k$ previously associated. Do
   $S_{k+1} = S_k \cup D_k$.
5: Do $k = k + 1$ and go to step 2.
```

The building of smimp for a normal proof, unlike of mimp-graph, is in the upwards direction, from conclusion to premises. If during the building of the proof we find a similar formula to a conclusion already derived, similar in the sense shown in the Algorithm 1. Instead of building two new branches for each of the similar formulas, we proceed as in the construction of the Definition 20 by induction. In the smimp the R-nodes ($\to$I, $\to$Iv, $\to$E) inside boxes may be shared any number of times, they represent rules with different inference orders. In the definition we add the item named “share” that describes how sharing is performed. The item “box” allows to add boxes and therefore distinguish between shared and unshared R-nodes.

Definition 20 (Smimp) *A smimp G is a directed graph* $\langle \mathrm{V}, \mathrm{E}, \mathrm{L}, (\mathrm{Box}_i)_{i\in I}\rangle$ *where:* V *is a set of nodes,* L *is a set of labels, E is a set of labeled* edges $\langle v \in\mathrm{V},\ t \in\text{E-Labels},\ v' \in\mathrm{V}\rangle$ *of source v, target v′ and label t and is identified with the arrow* $v\xrightarrow{t}v'$. $(\mathrm{Box}_i)_{i\in I}$ *is a collection of set of nodes of G, called the boxes. Moreover, the boxes* $(\mathrm{Box}_i)_{i\in I}$ *should be non-overlapping, two boxes are disjoint or one is contained in the other:* $\forall i,j \in I\ (\mathrm{Box}_i \cap \mathrm{Box}_j = \varnothing \vee \mathrm{Box}_i \subset \mathrm{Box}_j \vee \mathrm{Box}_j \subset \mathrm{Box}_i)$.

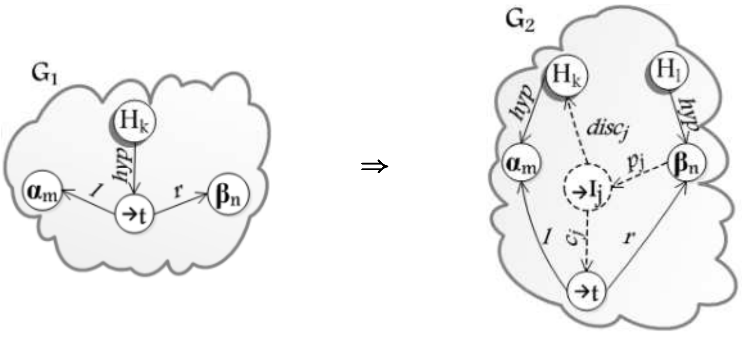
A smimp is defined recursively as follows:

Basis *If* G_1 *is a formula graph with root node* α_m[1], *then the graph* G_2 *that is defined as* G_1 *with the D-nodes* H_n *and C and the edges* $\alpha_m\xrightarrow{conc}C$ *and* $H_n\xrightarrow{hyp}\alpha_m$, *is a smimp.*

$\to$I *If* G_1 *is a smimp and contains the F-node* $\to_t$ *linked to the nodes* α_m, β_n *and* H_k *by the edges* $\to_t\xrightarrow{l}\alpha_m$, $\to_t\xrightarrow{r}\beta_n$ *and* $H_k\xrightarrow{l}\to_t$ *respectively, then the graph* G_2 *that is defined as* G_1 *with*

1. *the removal of the edge* $H_k\xrightarrow{hyp}\to_t$;
2. *an R-node* $\to\mathrm{I}_j$ *at the top position;*
3. *a D-node* H_l *linked to the F-node* β_n;
4. *the edges:* $H_k\xrightarrow{hyp}\alpha_m$, $\beta_n\xrightarrow{p_j}\to\mathrm{I}_j$, $\to\mathrm{I}_j\xrightarrow{c_j}\to_t$, $\to\mathrm{I}_j\xrightarrow{disc_j}H_k$;

is a smimp (see figure below; the α_m*-node is* discharged*).*

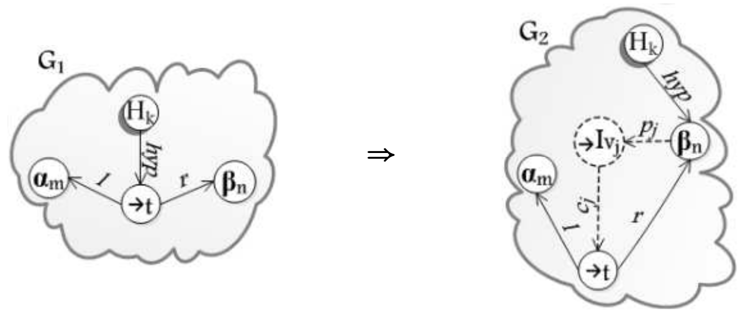


[1]We will use the terms α_m, β_n and γ_r to represent the principal connective of the formula α, β and γ respectively.

$\rightarrow$Iv [2] *If G_1 is a smimp and contains the F-node $\rightarrow_t$ linked to the nodes α_m, β_n and H_k by the edges $\rightarrow_t \xrightarrow{l} \alpha_m$, $\rightarrow_t \xrightarrow{r} \beta_n$ and $H_k \xrightarrow{hyp} \rightarrow_t$ respectively, then the graph G_2 that is defined as G_1 with*

1. *the removal of the edge $H_k \xrightarrow{hyp} \rightarrow_t$;*
2. *an R-node $\rightarrow\mathrm{Iv}_j$ at the top position;*
3. *a D-node H_k linked to the F-node β_n;*
4. *the edges: $H_k \xrightarrow{hyp} \beta_n$, $\beta_n \xrightarrow{p_j} \rightarrow\mathrm{Iv}_j$ and $\rightarrow\mathrm{Iv}_j \xrightarrow{c_j} \rightarrow_t$;*

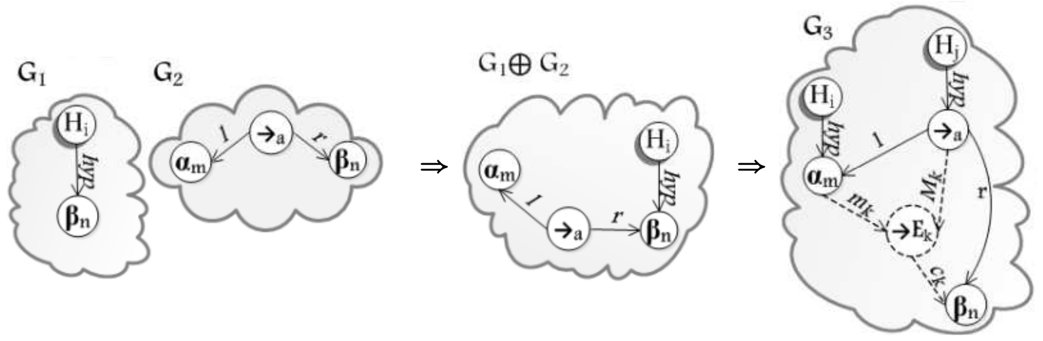
is a smimp (see figure below).



$\rightarrow$E *If G_1 is a smimp and G_2 is a formula graph with root node $\rightarrow_a$ linked to the nodes α_m, β_n by edges l and r, and the graph (intermediate step) obtained by $G_1 \oplus G_2$ contains the node β_n linked to the D-node H_i, then the graph G_3 that is defined as $G_1 \oplus G_2$ with*

1. *the removal of the edge $H_i \xrightarrow{hyp} \beta_n$;*
2. *an R-node $\rightarrow\mathrm{E}_k$ at the top position;*
3. *the edges: $H_i \xrightarrow{hyp} \alpha_m$, $H_j \xrightarrow{hyp} \rightarrow_a$, $\alpha_m \xrightarrow{m_k} \rightarrow\mathrm{E}_k$, $\rightarrow_a \xrightarrow{M_k} \rightarrow\mathrm{E}_k$ and $\rightarrow\mathrm{E}_k \xrightarrow{c_k} \beta_n$;*

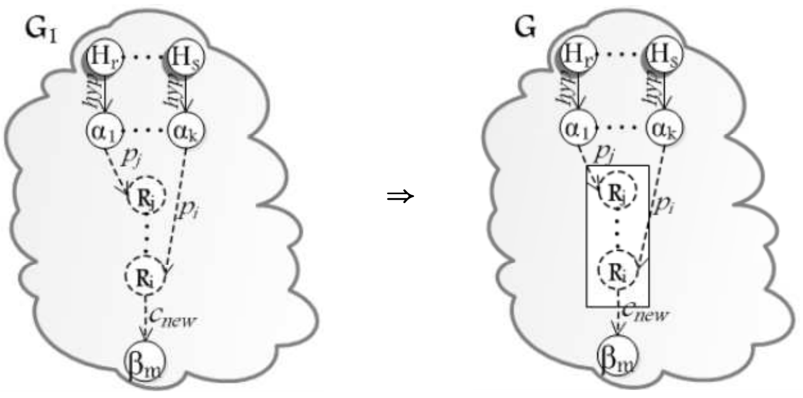
is a smimp (see figure below).



Box *If G_1 is a smimp and contains a sequence of R-nodes $R_i, ..., R_j$ that starts in the inference order i and ends in the inference order j, and this*

[2]the "v" stands for "vacuous"

sequence has zero or more premises $\alpha_1, ..., \alpha_k$ and one conclusion β_m, then the graph G that is defined as G_1 with a rule box Box= $\{R_i, ..., R_j\}$ *is a smimp (see figure below).*

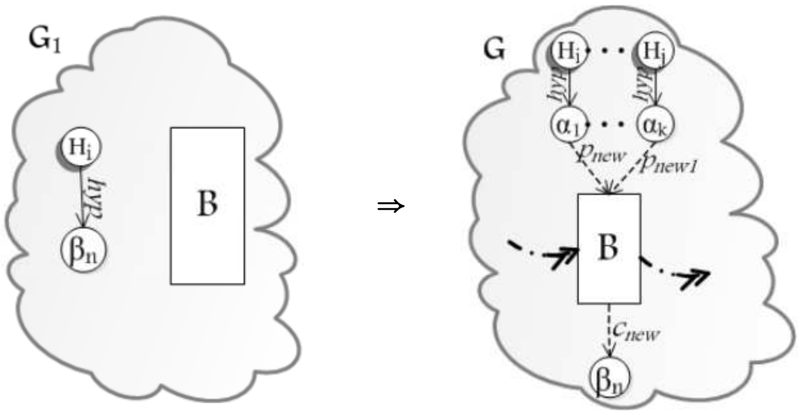


Share *If G_1 is a smimp containing the F-node β_n linked to a D-node H and there is a rule box* Box = $\{R_1, ..., R_m\}$ *and β_n is unifiable*[3] *with some conclusion graph of the box*[4] *and each element in $\alpha_1, ..., \alpha_k$ (desirable hypothesis) is unifiable with each premise*[5] *of the box, respectively. The graph G that is defined as $G_1 \oplus \alpha_1 \oplus ... \oplus \alpha_k$ with*

1. *the removal of the edge $H \xrightarrow{hyp} \beta_n$ and the D-node H;*
2. *the premises $\alpha_1 ... \alpha_k$ that are not associated with a D-node H will be associated with a new one;*
3. *for each R-node R_i in* Box
   - (a) *a new inferential order conserving the list of original orders given by: $(R_i/\ [o \mid new])$;*
   - (b) *for each premise F of R_i: apply σF*[6] *and add one edge (type p, m or M with index new) from σF to R_i;*
   - (c) *for the conclusion C of R_i: apply σC add one edge (type c with index new) from R_i to σC;*
   - (d) *if R_i has any discharged formula F then: apply σF and add one D-node H and one disc-edge (with index new) to σF;*

is a smimp (see figure below).

[3]in the sense shown in the Algorithm 1
[4]It is enough to compare at least one occurrence of them.
[5]It is enough to compare at least one occurrence of each premise.
[6]σ was generated by the Algorithm 1



Lemma 3 enables us to prove that a given graph G is a smimp without explicitly supplying a construction. The Lemma basically says that we just have to check that G has an inferential ordering on all R-nodes and that each node of G is of one of the possible types (P, K, E, I, B, H and C) that generate the construction cases of Definition 20.

Lemma 3 *G is a smimp if and only if the following hold*

1. *There exists a well-founded (hence acyclic) inferential order $<$ on all R-nodes of the smimp.*
2. *Every node N of G is of one of the following seven types:*

P *N is labeled with one of the propositional letters: $\{p,\ q,\ r,\ \ldots\ \}$. N has no outgoing edges l and r.*

K *N has label $\rightarrow_n$ and has exactly two outgoing edges with label l and r, respectively. N may has outgoing edges with labels p_i, m_j or M_k; and ingoing edges with label c_l and hyp_m.*

E *N has label $\rightarrow\mathrm{E}_k$ and has one (or n if the node is in a box) outgoing edges $\rightarrow\mathrm{E}_k \xrightarrow{c_k} \beta_n$, where β_n is a node type* **P** *or* **K**. *N has exactly two (or $2n$ if the node is in a box) ingoing edges: $\alpha_m \xrightarrow{m_k} \rightarrow\mathrm{E}_k$ and $\rightarrow_q \xrightarrow{M_k} \rightarrow\mathrm{E}_k$, where α_m is a node type* **P** *or* **K**. *there are also more exactly two outgoing edges from the node $\rightarrow_a$: $\rightarrow_a \xrightarrow{l} \alpha_m$ and $\rightarrow_a \xrightarrow{r} \beta_n$.*

I *N has label $\rightarrow\mathrm{I}_j$ (or $\rightarrow\mathrm{Iv}_j$, if it discharges an hypothesis vacuously), has one (or n if the node is in a box) outgoing edge $\rightarrow\mathrm{I}_j \xrightarrow{c_j} \rightarrow_t$), and one (or zero for the vacuous case $\rightarrow\mathrm{Iv}$, or n if the node is in a box) outgoing edge ($\rightarrow\mathrm{I}_j$, $disc_j$, H_k). N has exactly one (or n if the node is in a box) ingoing edge: $\beta_n \xrightarrow{p_j} \mathrm{I}_j$, where β_n is a node type* **P** *or* **K**. *There are two outgoing edges from the node $\rightarrow_t$: $\rightarrow_t \xrightarrow{l} \alpha_m$ and $\rightarrow_t \xrightarrow{r} \beta_n$ such that there is one (or zero for the case $\rightarrow\mathrm{Iv}$) ingoing edge to the node α_m: $H_k \xrightarrow{hyp} \alpha_m$.*

B *N is a rule box* Box *that contains R-nodes and each R-node can store one or more inferential orders and satisfy the property of non-overlap, two boxes are disjoint or one is contained in the other.*

H *N has label H_k and has outgoing edges with label hyp$_k$.*

C *N has label* C *and has exactly one ingoing edge with label conc.*

Proof: Similar to the proof of Lemma 1. ■

4.2 Discussion on R-minimal representation

There is a notion of "minimal representation" in the amount of F-nodes. We would like to speak about "minimal representation" in the amount of R-nodes but this has not yet been established because in the definition of graph has not been implemented a way to verify if all rule boxes occurring in S_M denote pairwise distinct boxes. So the Lemma proof below would be unfinished:

Lemma 4 *Every mimp-like representation M has a uniquely determined (up to graph-isomorphism)* R-minimal smimp-like representation S_M*, i.e. a representation satisfies the following four conditions.*

1. *S_M is a smimp whose size does not exceed the size of M.*
2. *M and S_M both have the same (set of) hypotheses and the same conclusion.*
3. *There is a graph homomorphism $h : M \rightarrow S_M$ that is injective on F-Labels.*
4. *All rule boxes occurring in S_M denote pairwise distinct boxes.*

4.3 Example of application

Consider a generalization φ_k detailed below for what we have the following fact from (Haeusler in press):

Proposition 2 *Any normal proof of φ_k in $M^{\rightarrow}$ has at least 2^k occurrences of the same assumptions, that are discharged by the last rule of the proof.*

The φ_k family of formulas can be defined as follows:

Definition 21 *Let $\chi[X,Y] = (((X \rightarrow Y) \rightarrow X) \rightarrow X) \rightarrow Y$. Using $\chi[X,Y]$ we recursively define a family of formulas. Consider the propositional letters C, D_k, $k > 0$, be the formula recursively defined as:*

$$\xi_1 = \chi[D_1, C] \tag{4-1}$$

$$\xi_{k+1} = \chi[D_{k+1}, \xi_k] \tag{4-2}$$

Using this family of formulas we define the formula φ_n, $n > 0$, such that, for any $k \geq 0$:

$$\varphi_{k+1} = \xi_{k+1} \to C$$

The following is a derivation of C from $(((d_{k+1} \to \xi_k) \to d_{k+1}) \to d_{k+1}) \to \xi_k$, and hence, by an $\to$-introduction we have a normal derivation of φ_{k+1}.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[d_1]^1}{((d_1 \to c) \to d_1) \to d_1}\,v \qquad \begin{matrix}[(((d_{k+1} \to \xi_k) \to d_{k+1}) \to d_{k+1}) \to \xi_k]^y \\ \Pi^* \\ \xi_1\end{matrix}}{c}}{d_1 \to c}\,1 \qquad [(d_1 \to c) \to d_1]^x}{d_1}}{((d_1 \to c) \to d_1) \to d_1}\,x \qquad \begin{matrix}[(((d_{k+1} \to \xi_k) \to d_{k+1}) \to d_{k+1}) \to \xi_k]^y \\ \Pi^* \\ \xi_1\end{matrix}}{c}}{((((d_{k+1} \to \xi_k) \to d_{k+1}) \to d_{k+1}) \to \xi_k) \to c}\,y$$

Using smimp, we can build a proof with unified parts which is much more economical than mimp-graph. Below we can find a table comparing both versions of mimp-graph when used to prove the class of formulas φ_k that have this exponential growth. A comparative table is presented in Table 4.1 and smimp representation in the Figure 4.4.

Linearized Mimp-Graph	**mimp-graph**	**smimp**
$l(G_1) = 69$	$l(G'_1) = 7 + 7 + Hyp(G'_1) = 17$, $Hyp(G'_1) = 3$	$l(G^c_1) = 7 + 7 + Hyp(G^c_1) = 17$, $Hyp(G^c_1) = 3$
$l(G_2) = 327$	$l(G'_2) = 19 + 12 + Hyp(G'_2) = 44$, $Hyp(G'_2) = 7$	$l(G^c_2) = 13 + 12 + Hyp(G^c_2) = 30$, $Hyp(G^c_2) = 5$
$l(G_3) = 1380$	$l(G'_3) = 43 + 17 + Hyp(G'_3) = 48$, $Hyp(G'_3) = 15$	$l(G^c_3) = 19 + 17 + Hyp(G^c_3) = 43$, $Hyp(G^c_3) = 7$
$l(G_4) =$	$l(G'_4) = 91 + 22 + Hyp(G'_4) = 144$, $Hyp(G'_4) = 31$	$l(G^c_4) = 25 + 22 + Hyp(G^c_4) = 56$, $Hyp(G^c_4) = 2 + Hyp(G^c_3)$
$\vdots$	$\vdots$	$\vdots$
$l(G_k) =$	$l(G'_k) = (2^k 6 - 5) + (2 + 5k) + Hyp(G'_k)$, $Hyp(G'_k) = 2^k + Hyp(G'_{k-1})$	$l(G^c_k) = (6k + 1) + (2 + 5k) + Hyp(G^c_k)$, $Hyp(G^c_k) = 2 + Hyp(G^c_{k-1})$
The length is given by $l(G) = rn + fn + Hyp(G)$, l: length, rn: number of R-nodes, fn: number of F-nodes, Hyp: number of D-nodes H		

Table 4.1: Comparative size of proofs

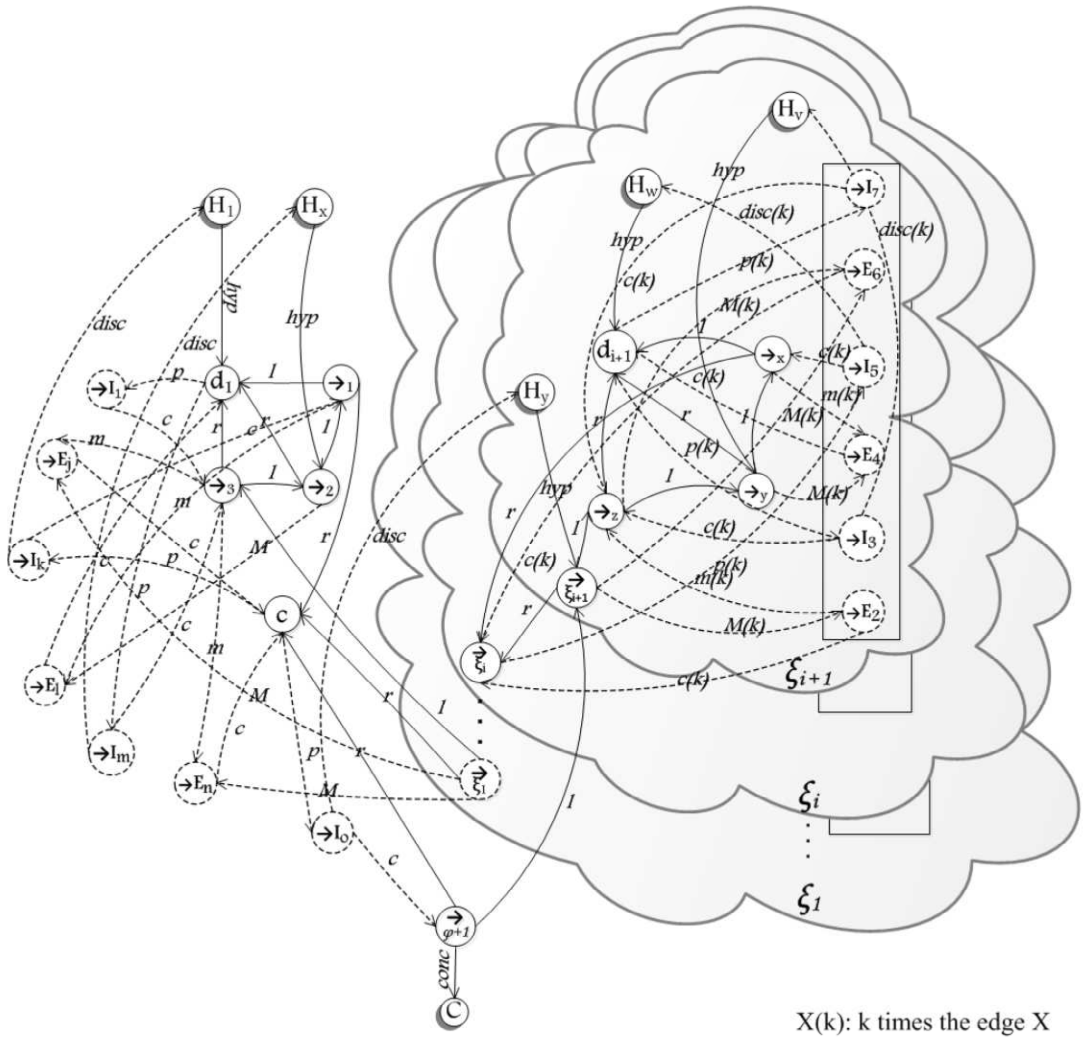


Figure 4.4: Smimp of φ_{k+1}

4.4 Conclusion

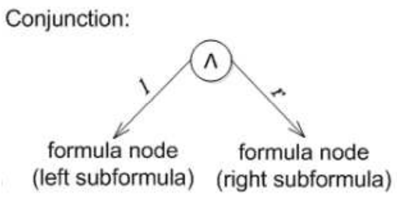
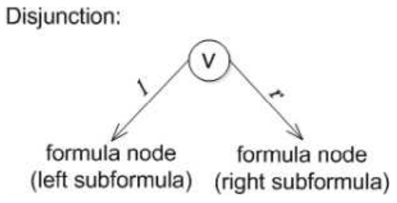
The treatment for inference rules, in addition to formula sharing, is performed during the construction process of the graph. This feature is of fundamental importance, since we intend to use this graph in automatic theorem provers. In the construction process when similar formulas are found, we share a sub-proof and produce the unifier in linear time. We do not implement a process of searching for similar formulas in our graph but we estimate that the resources consumed in such searching would be compensated by the reduction of necessary resources to build the proof-graph.

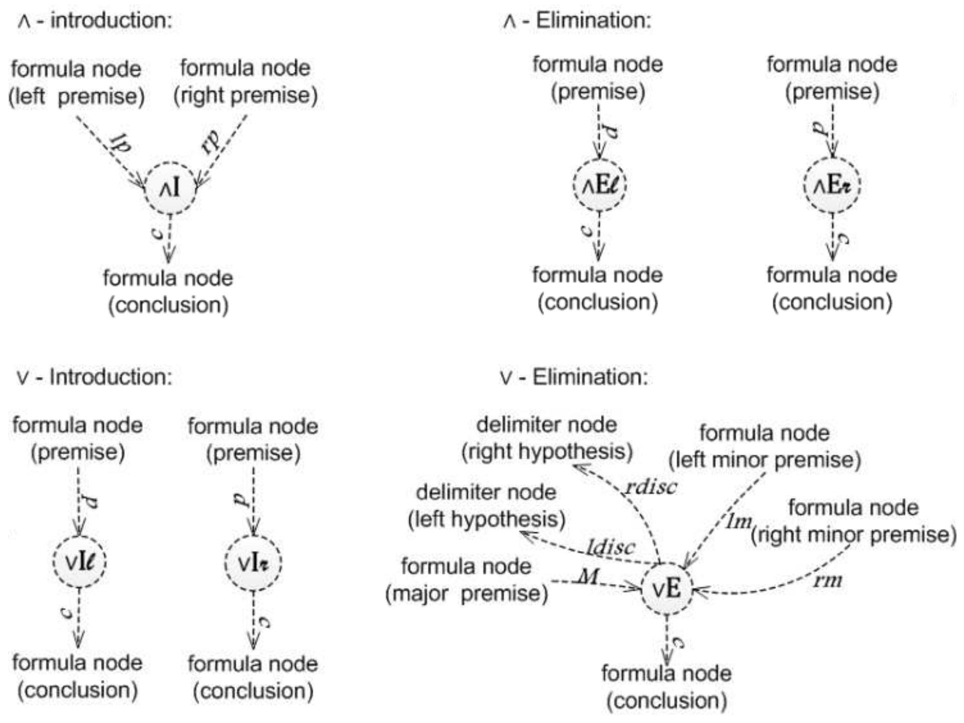
5 Extending mimp-graphs

Mimp-graphs provide a formalism for Natural Deduction where the use of a "mixed" graph representation of formulas and inferences (in the purely implicational minimal logic) serve as a way to study the complexity of proofs and to provide more efficient theorem provers. We presented the main notion of mimp-graphs in Chapter 3. In this chapter, we wish to extent this formalism in graphs for full minimal propositional calculus and for first order logic.

5.1 Proof-graphs for propositional logic

In Chapter 3 we considered implication as the only logic connective. Let us now turn to a more general presentation of proof-graphs for full minimal propositional logic that includes $\rightarrow$ (implication), $\vee$ (disjunction) and $\wedge$ (conjunction). We also develop the normalization procedure for these proof-graphs. Mimp-graphs for propositional logic will be defined along with partial ordering on its R-nodes that allows to pass through the nodes of the structure. We will also develop the normalization procedure for these proof-graphs.

- the set of formula labels *F-Labels* in Definition 5 has two added labels $\vee$, $\wedge$;
- the set of inference labels *R-Labels* has the added labels: {$\wedge$I, $\wedge$El, $\wedge$Er, $\vee$Il, $\vee$Ir, $\vee$E};
- the set of edge labels E_M*-Labels* has the added labels: {lp (left premise), rp (right premise), lm (left minor premise), rm (right minor premise) , $ldisc$ (discharge to the left), $rdisc$ (discharge to the right)};
- the definition of formula graphs has two added inductive graphs, i.e. 
Conjunction:
∧
l
r
formula node (left subformula)
formula node (right subformula)

Disjunction:
∨
l
r
formula node (left subformula)
formula node (right subformula)
- inference rules $\wedge$-Introduction, $\wedge$-Elimination, $\vee$-Introduction and $\vee$-Elimination in proof-graphs are as follows:


∧ - introduction:
formula node (left premise)
formula node (right premise)
lp
rp
∧I
c
formula node (conclusion)
∧ - Elimination:
formula node (premise)
p
∧Eℓ
c
formula node (conclusion)
formula node (premise)
p
∧Er
c
formula node (conclusion)
∨ - Introduction:
formula node (premise)
p
∨Iℓ
c
formula node (conclusion)
formula node (premise)
p
∨Ir
c
formula node (conclusion)
∨ - Elimination:
delimiter node (right hypothesis)
formula node (left minor premise)
rdisc
delimiter node (left hypothesis)
lm
formula node (right minor premise)
ldisc
formula node (major premise)
M
∨E
rm
c
formula node (conclusion)

In the terminology about inference rules or R-nodes, when an R-node has more than one incoming edge, these are distinguished by calling them left, right, major or minor, or a combination of these terms and so also the F-node 'premise' associated with these edges. Thus, the major premise in R-node contains the connective that is eliminated; the other premise in R-node is called 'minor'. Two premises that play a more or less equal role in the inference are called 'left' and 'right'. For instance, an R-node ∨E has a major premise, a left minor premise and a right minor premise; an R-node ∧I has a left premise and a right premise.

The term *R-node sequence* is representing a deduction, and if it is a smaller part of another R-node sequence (subdeduction), then it is called *a subsequence* of the latter. A subsequence that derives a premise of the last R-node application in an R-node sequence is called a direct R-node subsequence. Instead of writing "the direct R-node subsequence that derives the minor premise of the last inference of an R-node sequence D", we simply write *"the minor subsequence of D".*

Definition 22 *Let* $G_1 = \langle V^1, E^1, L^1 \rangle$ *and* $G_2 = \langle V^2, E^2, L^2 \rangle$ *be two graphs, where:* V^1 *and* V^2 *are sets of vertices,* E^1 *and* E^2 *are sets of labeled* edges, L^1 *and* L^2 *are subsets of* LBL. *The operation* $G_1 \odot G_2 := \langle V^1 \sqcup V^2, E^1 \not\ddagger E^2, L^1 \cup L^2 \rangle$ *equalizes R-nodes of* G_1 *with R-nodes of* G_2 *that have the same set of premises and conclusion keeping the inferential order of each node, and equalizes F-nodes of* G_1 *with F-nodes of* G_2 *that have the same label, and equalizes edges with the same source, target and label into one.*

Definition 23 *A* mimp-graph *for propositional logic G is a directed graph* ⟨V, E, L⟩ *where:* V *is a set of nodes,* L *is a set of labels,* E *is a set of labeled* edges ⟨$v \in$ V, $t \in$ L, $v' \in$ V⟩, *of source v, of target v' and label t.*

Propositional mimp-graphs are recursively defined as follows:

mimp *Every construction rule for mimp-graphs (Definition 9) is a construction rule for propositional mimp-graph.*

$\wedge$I *If G_1 and G_2 are propositional mimp-graphs and G_1 contains α_m linked to the D-node C and G_2 contains β_n linked to the D-node C, then the graph G that is defined as*

1. *$G := G_1 \oplus G_2 \oplus G_3$ with the removal of ingoing edges in the node C which were generated in the intermediate step (see Figure 5.1, dotted area in $G_1 \oplus G_2$);*
2. *an R-node $\wedge \mathrm{I}_i$ at the top position;*
3. *edges: $\alpha_m \xrightarrow{lp_i} \wedge \mathrm{I}_i$, $\beta_n \xrightarrow{rp_i} \wedge \mathrm{I}_i$, $\wedge \mathrm{I}_i \xrightarrow{c_i} \wedge_t$ and $\wedge_t \xrightarrow{conc} C$,*

is a mimp-graph (see Figure 5.1).

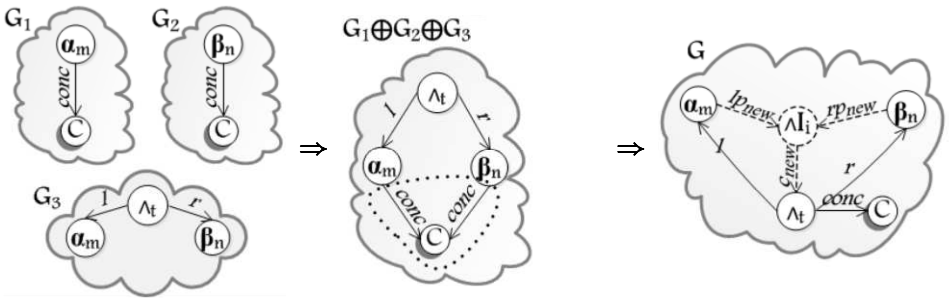


Figure 5.1: The $\wedge$I rule of the propositional mimp-graph.

$\wedge$El *If G_1 is a propositional mimp-graph and contains edges $\wedge_t \xrightarrow{l} \alpha_m$, $\wedge_t \xrightarrow{r} \beta_n$) and the node $\wedge_t$ linked to the D-node C then the graph G that is defined as G_1 with*

1. *the removal of the ingoing edge in the node C;*
2. *an R-node $\wedge \mathrm{E}l_i$ at the top position;*
3. *edges: $\wedge_t \xrightarrow{p_i} \wedge \mathrm{E}l_i$, $\wedge \mathrm{E}l_i \xrightarrow{c_i} \alpha_m$ and $\alpha_m \xrightarrow{conc} C$;*

is a mimp-graph. There is a symmetric case for $\wedge Er$.

$\vee$Il *If G_1 is a propositional mimp-graph and contains nodes α_m linked to the D-node C then the graph G that is defined as G_1 with*

1. *the removal of the ingoing edge in the node C.*

2. *an R-node* $\vee\mathrm{I}l_i$ *at the top position;*
3. *edges:* $\vee\mathrm{I}l_i \xrightarrow{c_i} \vee_t$, $\alpha_m \xrightarrow{p_i} \vee\mathrm{I}l_i$, $\vee_t \xrightarrow{l} \alpha_m$, $\vee_t \xrightarrow{r} \beta_n$ *and* $\vee_t \xrightarrow{conc} C$,

is a mimp-graph. There is a symmetric case for $\vee\mathrm{I}r$.

$\vee$**E** *If* G_1, G_2 *and* G_3 *are propositional mimp-graphs, and the graph obtained by* $(G_1 \odot G_2) \oplus G_3$ *(intermediate step) contains nodes:* $\vee_t$ *and* σ_r *linked to the D-node* C *(*σ_r *twice); and* α_m *and* β_n *are subformulas of* $\vee_t$ *and are linked to D-nodes* H*, then the graph* G *that is defined as* $(G_1 \odot G_2) \oplus G_3$ *with*

1. *the removal of ingoing edges in the node* C *which were generated in the intermediate step (see Figure 5.2);*
2. *an R-node* $\vee E_i$ *at the top position;*
3. *edges:* $\sigma_r \xrightarrow{lm_i} \vee\mathrm{E}_i$, $\sigma_r \xrightarrow{rm_i} \vee\mathrm{E}_i$, $\vee_t \xrightarrow{M_i} \vee\mathrm{E}_i$, $\vee\mathrm{E}_i \xrightarrow{c_i} \sigma_r$, $\vee\mathrm{E}_i \xrightarrow{ldisc_i} H_u$, $\vee\mathrm{E}_i \xrightarrow{rdisc_i} H_s$ *and* $\sigma_r \xrightarrow{conc} C$,

is a mimp-graph (see Figure 5.2).

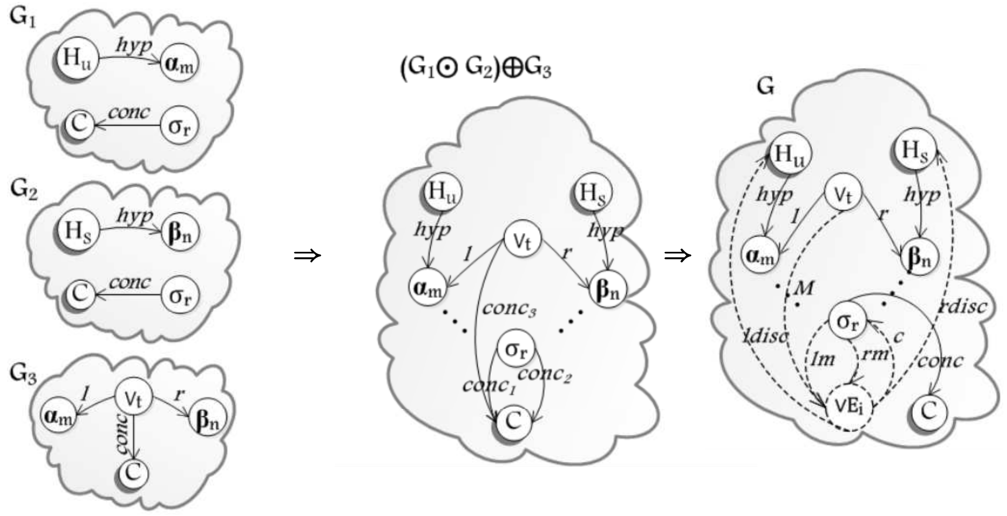


Figure 5.2: The $\vee$E rule of the propositional mimp-graph.

Lemma 5 enables us to prove that a given graph G is a propositional mimp-graph without explicitly supplying a construction. Among others it says that we have to check that each node of G is of one of possible types that generate the construction cases of Definition 23.

In order to avoid overloading of indexes, we will omit whenever possible, the indexing of edges of kind *lm*, *rm*, *lp*, *rp*, *ldisc* and *rdisc*, keeping in mind that the coherence of indexing is established by the kind of rule-node to which they are linked.

Lemma 5 *G is a propositional mimp-graph if and only if the following hold:*

1. *There exists a well-founded (hence acyclic) inferential order $<$ on all rule nodes of the propositional mimp-graph.*

2. *Every node N of G is of one of the following ten types:*

P *It is as in the Lemma 1.*

F *N has one of the following labels: $\rightarrow_i$, $\wedge_j$ or $\vee_k$, and has exactly two outgoing edges with label l and r. N has outgoing edges with labels p, m, M, lm, rm, lp, rp; and ingoing edges with label c and hyp.*

I$^\wedge$ *N has label $\wedge\mathrm{I}_i$, one outgoing edge $\wedge\mathrm{I}_i \xrightarrow{c} \wedge_t$ and exactly two ingoing edges: $\alpha_m \xrightarrow{lp} \wedge I_i$ and $\beta_n \xrightarrow{rp} \wedge\mathrm{I}_i$, where α_m and β_n are nodes type* **P** *or* **F***. There are two outgoing edges from the node $\wedge_t$: $\wedge_t \xrightarrow{l} \alpha_m$ and $\wedge_t \xrightarrow{r} \beta_n$.*

E$^\wedge$ *N has label $\wedge\mathrm{E}_i$, one outgoing edge $\wedge\mathrm{E}l_i \xrightarrow{c} \alpha_m$ where α_m (or β_n in the case $\wedge\mathrm{E}r_i$ is a node type* **P** *or* **F** *and has exactly one ingoing edge: $\wedge_t \xrightarrow{p} \wedge\mathrm{E}_i$. There are two outgoing edges from the node $\wedge_t$: $\wedge_t \xrightarrow{l} \alpha_m$ and $\wedge_t \xrightarrow{r} \beta_n$.*

I$^\vee$ *N has label $\vee\mathrm{I}l_i$, one outgoing edge $\vee\mathrm{I}l_i \xrightarrow{c} \vee_t$ and has exactly one ingoing edge: $\alpha_m \xrightarrow{p} \vee\mathrm{I}l_i$ where α_m (or β_n in the case $\vee\mathrm{I}r_i$) is a node type* **P** *or* **F***. There are two outgoing edges from the node $\vee_t$: $\vee_t \xrightarrow{l} \alpha_m$ and $\vee_t \xrightarrow{r} \beta_n$.*

E$^\vee$ *N has label $\vee E_i$, three outgoing edges $\vee\mathrm{E}_i \xrightarrow{c} \sigma_r$, $\vee\mathrm{E}_i \xrightarrow{ldisc} H_u$ and $\vee\mathrm{E}_i \xrightarrow{rdisc} H_s$; and it has exactly three ingoing edges: $\vee_t \xrightarrow{M} \vee\mathrm{E}_i$, $\sigma_r \xrightarrow{lm} \vee E_i$, $\sigma_r \xrightarrow{rm} \vee\mathrm{E}_i$ where α_m (or β_n in the case $\vee\mathrm{E}_i$) is a node type* **P** *or* **F***. There are two outgoing edges from the node $\vee_t$: $\vee_t \xrightarrow{l} \alpha_m$, $\vee_t \xrightarrow{r} \beta_n$ and hypothesis edges: $H_u \xrightarrow{hyp} \alpha_m$ and $H_s \xrightarrow{hyp} \beta_n$.*

I, E, H, C *They are as in the Lemma 1.*

Proof:

$\Rightarrow$: Argue by induction on the construction of propositional mimp-graph (Definition 23). For every construction case for propositional mimp-graphs we have to check the three properties stated in Lemma. Property (2) is immediate. For property (1), we know from the induction hypothesis that there is an inferential order $<$ on R-nodes of the propositional mimp-graph. In the new construction cases $\wedge$I, $\wedge$El, $\wedge$Er, $\vee$Il, $\vee$Ir or $\vee$E, we make the new R-node that is introduced highest in the $<$-ordering, which yields an inferential ordering on R-nodes. In the construction case $\wedge$I, when we have two inferential orderings, $<_1$ on G_1 and $<_2$ on G_2. Then $G_1 \oplus G_2$ can be given an inferential ordering by taking the union of $<_1$ and $<_2$ and in addition putting $n < m$ for every R-node n, m such that $n \in G_1, m \in G_2$. In the construction case $\vee$E, when we have three inferential orderings, $<_1$ on G_1, $<_2$ on G_2 and $<_3$ on G_3. Then $(G_1 \odot G_2) \oplus G_3$ can be given an inferential ordering by taking the union of $<_1$, $<_2$ and $<_3$ and in addition putting $n < m < p$ for every R-node n, m, p such that $n \in G_1, m \in G_2, p \in G_3$.

$\Leftarrow$: Argue by induction on the number of R-nodes of G. Let $<$ be the topological order that is assumed to exist. Let n be the R-node that is maximal w.r.t. $<$. Then n must be on the top position. When we remove node n, including its edges linked (if n is of type I$^{\vee}$) and the node type C is linked to the premise of the R-node, we obtain a graph G' that satisfies properties listed in Lemma. By induction hypothesis we see that G' is a propositional mimp-graph. Now we can add the node n again, using one of construction cases for propositional mimp-graphs: *mimp* if n is a L node, F node, $\rightarrow$E node or $\rightarrow$I node, I$^{\wedge}$ if n is a $\wedge$I node, E$^{\wedge}$ if n is a $\wedge$El node or $\wedge$Er node, I$^{\vee}$ if n is a $\vee$Il node or $\vee$Ir node, E$^{\wedge}$ if n is a $\vee$E node. ■

5.2 Normalization for propositional mimp-graphs

5.2.1 Elimination of maximal formula

In this section, we describe the normalization process for propositional mimp-graphs. Eliminating a maximal formula is very similar to the procedure for mimp-graphs described in Chapter 3, where we considered only the case of implication, now we define maximal formulas in conjunction, disjunction and implication:

Definition 24 *A* maximal formula *m in a propositional mimp-graph G is a sub-graph of G as follows:*

- $\wedge$***I followed by*** $\wedge$El. *It is composed of (see Figure 5.3):*

  1. *F-nodes: α_m, β_n and $\wedge_q$, where $\wedge_q$ has zero or more ingoing/outgoing edges*[1]*, e.g. $\wedge_q$ could be premise or conclusion of others R-nodes;*
  2. *R-nodes: $\wedge I_i$ and $\wedge \mathrm{E}l_l$, where $\wedge I_i$ has an inferential order lower than $\wedge \mathrm{E}l_l$ and there are zero or more maximal formulas between them*[2]*. If*

[1]Represented in the figure by double-headed arrows
[2]The maximal formulas are represented in the figure by nodes labeled with I and E

these nodes occur in different branches, a branch must be insertable[3] *in the other branch or bifurcated by an R-node* $\vee E$;

3. *edges:* $\wedge_q \xrightarrow{l} \alpha_m$, $\wedge_q \xrightarrow{r} \beta_n$, $\alpha_m \xrightarrow{lp} \wedge I_i$, $\beta_n \xrightarrow{rp} \wedge I_i$, $\wedge I_i \xrightarrow{c} \wedge_q$, $\wedge_q \xrightarrow{p} \wedge \mathrm{E}l_l$ *and* $\wedge \mathrm{E}l_l \xrightarrow{c} \alpha_m$.

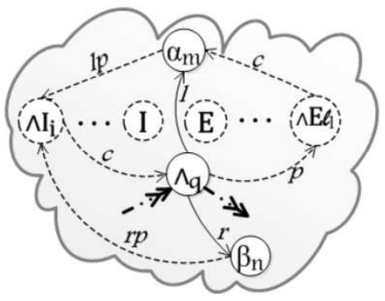


Figure 5.3: The maximal formula: $\wedge I$ followed by $\wedge El$.

There is a symmetric case for $\wedge I$ *followed by* $\wedge Er$.

– $\vee \mathrm{I}l$ ***followed by*** $\vee \mathbf{E}$. *It is composed of (see Figure 5.4):*

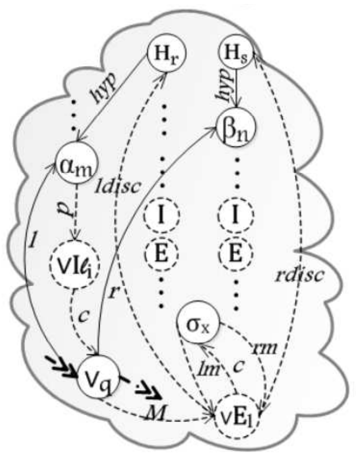


Figure 5.4: The maximal formula: $\vee Il$ followed by $\vee E$.

1. *F-nodes:* α_m, β_n, $\vee_q$ *and* σ_x, *where* $\vee_q$ *has zero or more ingoing/outgoing edges*[4];
2. *D-nodes:* H_r *and* H_s;
3. *R-nodes in ascending inferential order:* $\vee \mathrm{I}l_i$*and* $\vee E_l$, *and there are zero or more maximal formulas in branches between them*[5]. *If these nodes occur in different branches, a branch must be insertable in the other branch or bifurcated by an R-node* $\vee E$;

[3]A branch is insertable in other branch when it is bifurcated by a maximal formula: $\to$I followed by $\to$E

[4]Represented in the figure by double-headed arrows

[5]Maximal formulas are represented in the figure by nodes labeled with I and E

4. *edges:* $\vee_q \xrightarrow{l} \alpha_m$, $\vee_q \xrightarrow{r} \beta_n$, $\alpha_m \xrightarrow{p} \vee Il_i$, $\vee Il_i \xrightarrow{c} \vee_q$, $\vee_q \xrightarrow{M} \vee E_l$, $\sigma_x \xrightarrow{lm} \vee E_l$, $\sigma_x \xrightarrow{rm} \vee E_l$, $\vee E_l \xrightarrow{c} \sigma_x$, $\vee E_l \xrightarrow{ldisc} H_r$ *and* $\vee E_l \xrightarrow{rdisc} H_s$.

There is a symmetric case for $\vee$Ir *followed by* $\vee E$.

– **$\rightarrow$I *followed by* $\rightarrow$E.** *It is composed of (see Figure 5.5):*

1. *F-nodes:* α_m, β_n *and* $\rightarrow_q$, *where* $\rightarrow_q$ *has zero or more ingoing/outgoing edges*[6]*;*
2. *the D-node:* H_u*;*
3. *R-nodes in ascending inferential order:* $\rightarrow I_i$ *and* $\rightarrow E_l$, *and there are zero or more maximal formulas between them*[7]*. If these nodes occur in different branches, a branch must be insertable in the other branch or bifurcated by an R-node* $\vee E$*;*
4. *edges:* $\rightarrow_q \xrightarrow{l} \alpha_m$, $\rightarrow_q \xrightarrow{r} \beta_n$, $\beta_n \xrightarrow{p} \rightarrow I_i$, $\rightarrow I_i \xrightarrow{c} \rightarrow_q$, $\rightarrow I_i \xrightarrow{disc} H_u$, $\rightarrow_q \xrightarrow{M} \rightarrow E_l$, $\alpha_m \xrightarrow{m} \rightarrow E_l$ *and* $\rightarrow E_l \xrightarrow{c} \beta_n$.

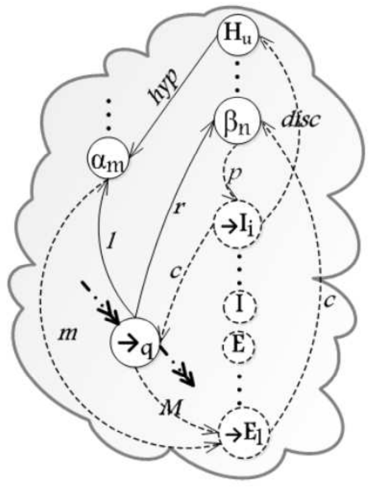


Figure 5.5: The maximal formula: $\rightarrow$I followed by $\rightarrow$E.

Definition 25 *The operation* incorporate *adds an R-node sequence inside other R-node sequence where it shares the same formula-graphs premise and conclusion, then apply the operation defined in Definition 13 to the resulting graph. Note that Proposition 1 ensures that the result is a mimp-graph.*

Note that the actual situation is more complicated than those sketched in Figures 5.3, 5.4 and 5.5. There are five sub-cases for each maximal formula due to the presence of disjunction and other maximal formulas. These sub-cases are treated in the Definition 26 as follows.

[6]Represented in the figure by double-headed arrows

[7]Maximal formulas are represented in the figure by nodes labeled with I and E

Definition 26 *Given a propositional mimp-graph G with a maximal formula m,* eliminating a maximal formula *is the following transformation of a propositional mimp-graph:*

Elimination of $\wedge I$ followed by $\wedge El$. *There is a symmetric case for the elimination of $\wedge I$ followed by $\wedge$Er. The elimination of the maximal formula $\wedge I$ followed by $\wedge$El is the following operation on a propositional mimp-graph:*

1. *If there are no* maximal formulas *between R-nodes* $\wedge \mathrm{I}_i$ *and* $\wedge \mathrm{El}_l$ *then follow these steps:*
   - (a) *If* $\wedge \mathrm{I}_i$ *and* $\wedge \mathrm{El}_l$ *are* not bifurcated *by one* $\vee$E *then (see cases 1 and 2 in Figure 5.6).*
     - i. *Remove R-nodes* $\wedge \mathrm{I}_i$ *and* $\wedge \mathrm{El}_l$ *and their edges.*
     - ii. *If the F-node* $\wedge_q$ *only has outgoing edges to sub-formulas then remove it (see case 2 in Figure 5.6).*

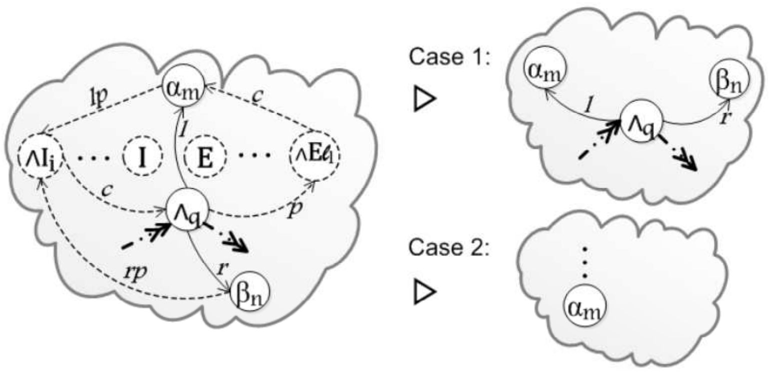


Figure 5.6: Elimination of $\wedge I$ followed by $\wedge El$: Cases 1 and 2.

   - (b) *Else If* $\wedge \mathrm{I}_i$ *represents two R-nodes then (see case 3 in Figure 5.7):*
     - i. *Remove the R-node* $\wedge \mathrm{I}_i$ *and its edges.*
     - ii. *Eliminate edges:* $\wedge_q \xrightarrow{lm} \vee \mathrm{E}_k$, $\wedge_q \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \wedge_q$.
     - iii. *If the F-node* $\wedge_q$ *only has outgoing edges to sub-formulas then remove it (see case 4 in Figure 5.7).*
     - iv. *Add edges:* $\alpha_m \xrightarrow{lm} \vee \mathrm{E}_k$, $\alpha_m \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \alpha_m$.

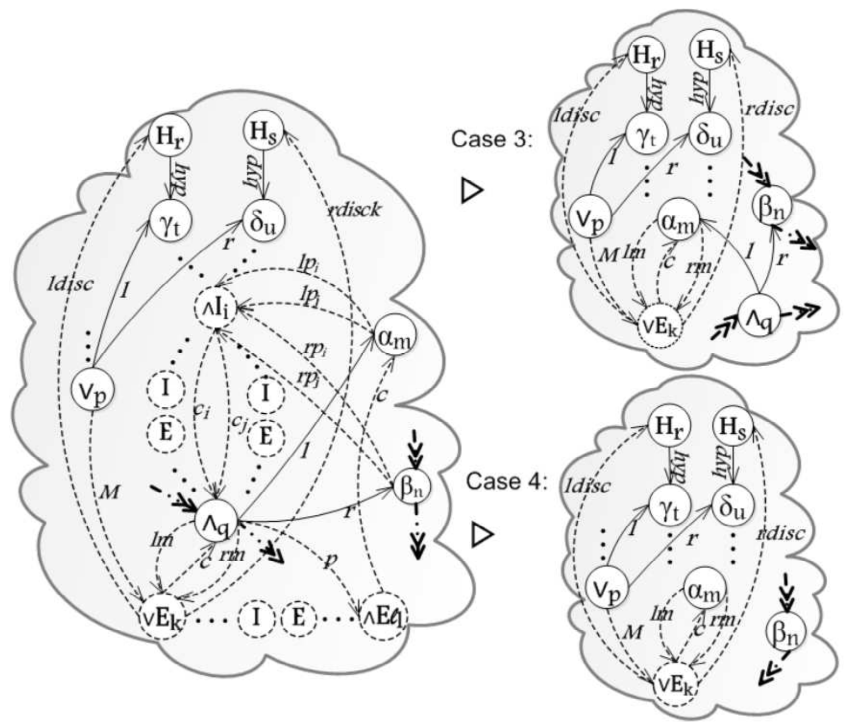


Figure 5.7: Elimination of $\wedge I$ followed by $\wedge El$: Cases 3 and 4.

(c) *Else (see case 5 in Figure 5.8)*

i. *Remove the R-node* $\wedge \mathrm{I}_i$*, and its edges.*

ii. *Eliminate edges:* $\wedge_q \xrightarrow{lm} \vee \mathrm{E}_k$*,* $\wedge_q \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \wedge_q$*.*

iii. *Add edges:* $\alpha_m \xrightarrow{lm} \vee \mathrm{E}_k$*,* $\alpha_m \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \alpha_m$*.*

iv. *Incorporate the R-node* $\wedge \mathrm{El}_l$*, as defined in Definition 25, in the right minor sequence of* $\vee \mathrm{E}_k$ *as last inference.*

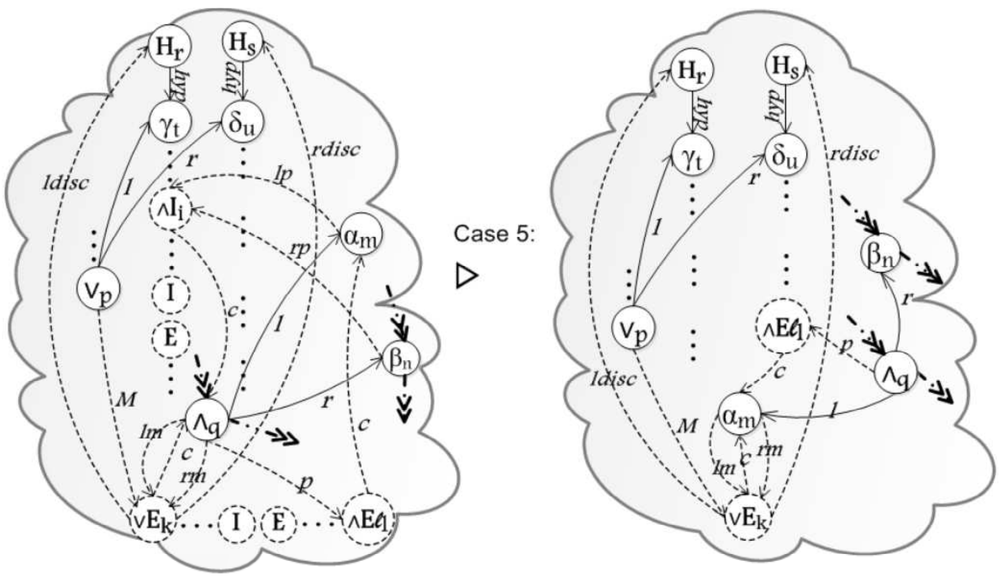


Figure 5.8: Elimination of $\wedge I$ followed by $\wedge El$: Case 5.

2. *Otherwise eliminate* maximal formulas *between R-nodes* $\wedge$I$_i$ *and* $\wedge$El_l.

Elimination of $\vee Il$ followed by $\vee E$ *There is a symmetric case for* $\vee$Ir *followed by* $\vee$E. *The elimination of this maximal formula is the following operation on a mimp-graph:*

1. *If there are no* maximal formulas *in branches between R-nodes* $\vee$Il_i *and* $\vee$E$_l$ *then follow these steps:*
   (a) *If* $\vee$Il_i *and* $\vee$E$_l$ *are* not bifurcated *by one* $\vee$E *then (see cases 1 and 2 in Figure 5.9).*
      i. *Remove R-nodes* $\vee$Il_i, $\vee$E$_l$, H$_r$ *and* H$_s$, *and their edges.*
      ii. *If the F-node* $\vee_q$ *only has outgoing edges to sub-formulas then remove it (see case 2 in Figure 5.9).*

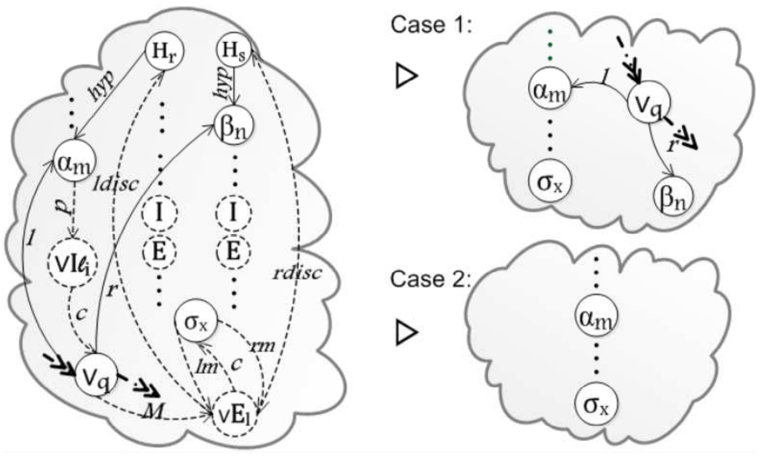


Figure 5.9: Elimination of $\vee Il$ followed by $\vee E$: Cases 1 and 2.

   (b) *Else If* $\vee$Il_i *represents two R-nodes then (see case 3 in Figure 5.10):*
      i. *Remove R-nodes* $\vee$Il_i, $\vee$E$_l$, H$_t$ *and* H$_u$, *and their edges.*
      ii. *Eliminate edges:* $\vee_q \xrightarrow{lm} \vee \mathrm{E}_k$, $\vee_q \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \vee_q$.
      iii. *If the F-node* $\vee_q$ *only has outgoing edges to sub-formulas then remove it (see case 4 in Figure 5.10).*
      iv. *Add edges:* $\sigma_x \xrightarrow{lm} \vee \mathrm{E}_k$, $\sigma_x \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \sigma_x$.
      v. *Incorporate the sequence* Π_x^m *of the Figure 5.10, as defined in Definition 25, in left and right minor subsequences of* $\vee$E$_k$.

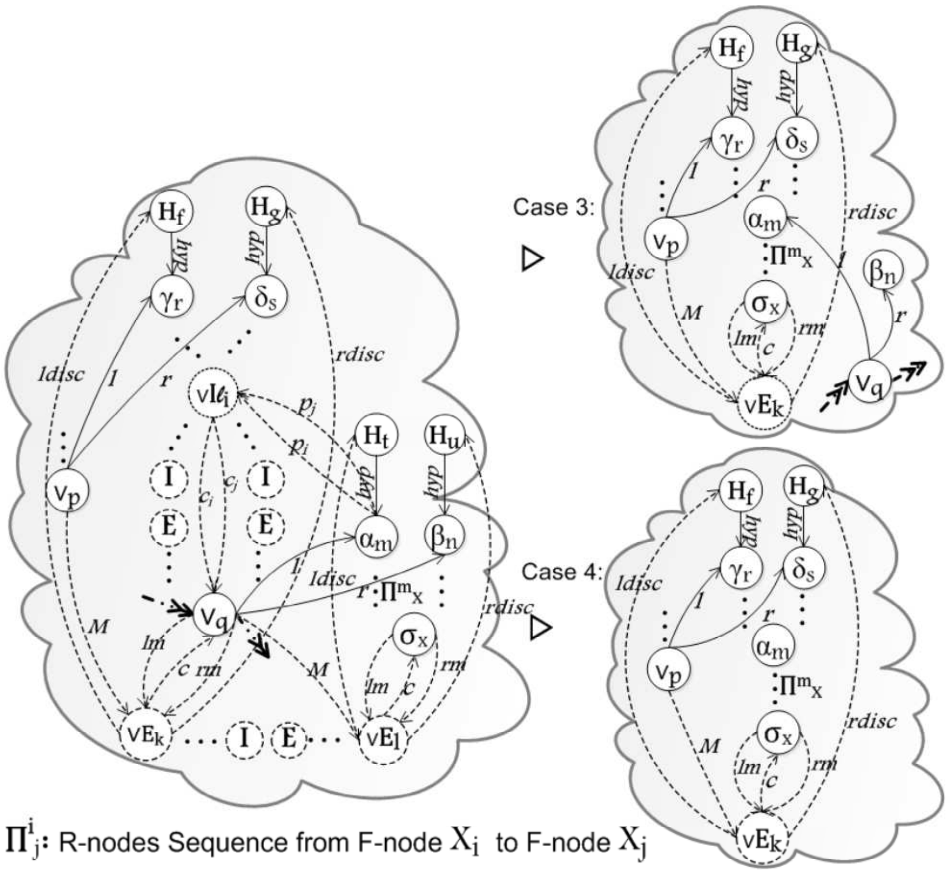

Case 3:
Case 4:
ldisc
rdisc
hyp
lm
rm
M
Π^i_j: R-nodes Sequence from F-node X_i to F-node X_j

Figure 5.10: Elimination of $\vee Il$ followed by $\vee E$: Cases 3 and 4.

(c) *Else (see case 5 in Figure 5.11)*

   i. *Remove the R-node* $\vee \mathrm{I}l_i$, *and its edges.*

   ii. *Eliminate edges:* $\vee_q \xrightarrow{lm} \vee \mathrm{E}_k$, $\vee_q \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \vee_q$.

   iii. *Add edges:* $\sigma_x \xrightarrow{lm} \vee \mathrm{E}_k$, $\sigma_x \xrightarrow{rm} \vee \mathrm{E}_k$ *and* $\vee \mathrm{E}_k \xrightarrow{c} \sigma_x$.

   iv. *Incorporate the R-node* $\vee \mathrm{E}_l$ *with its sub-sequences* Π^m_x *and* Π^n_x *showed in Figure 5.11, as defined in Definition 25, in the right minor subsequence of* $\vee \mathrm{E}_k$ *and incorporate the R-node sequence* Π^m_x *in the left minor premise of* $\vee \mathrm{E}_k$.

2. *Otherwise eliminate the* maximal formulas *in branches between R-nodes* $\vee \mathrm{I}l_i$ *and* $\vee E_l$.

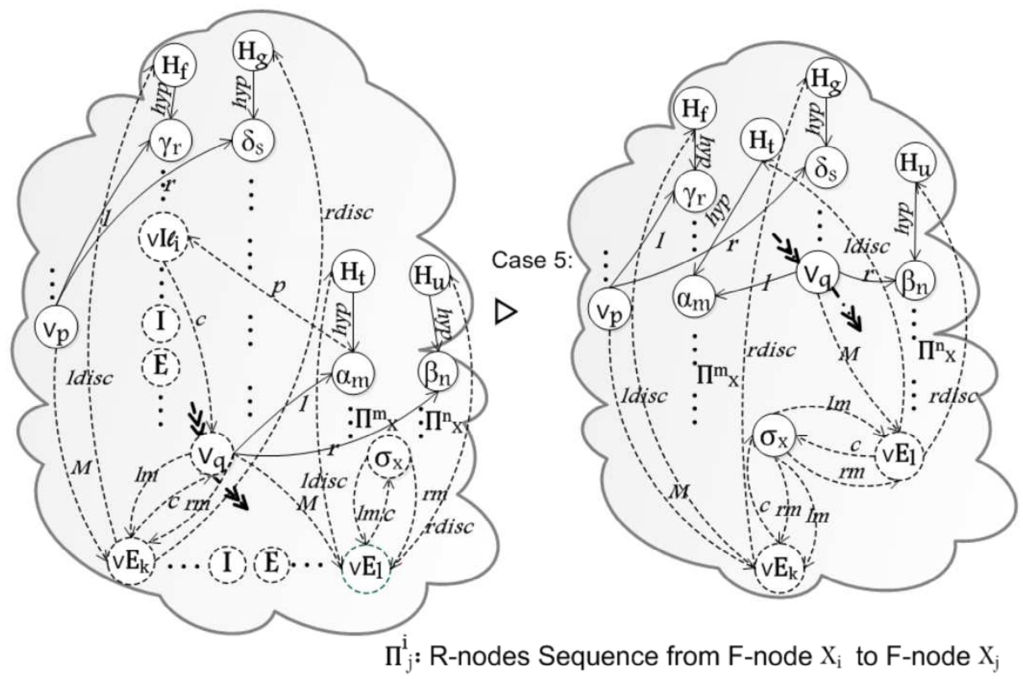


Figure 5.11: Elimination of $\vee Il$ followed by $\vee E$: Case 5.

Elimination of ➔*I* followed by ➔*E* *In order to reduce this, we need the following rule:*

1. *If there are no* maximal formulas *between R-nodes* ➔I$_i$ *and* ➔E$_l$ *then follow these steps:*

   (a) *If* ➔I$_i$ *and* ➔E$_l$ *are* not bifurcated *by one* ∨E *then (see cases 1 and 2 in Figure 5.12).*

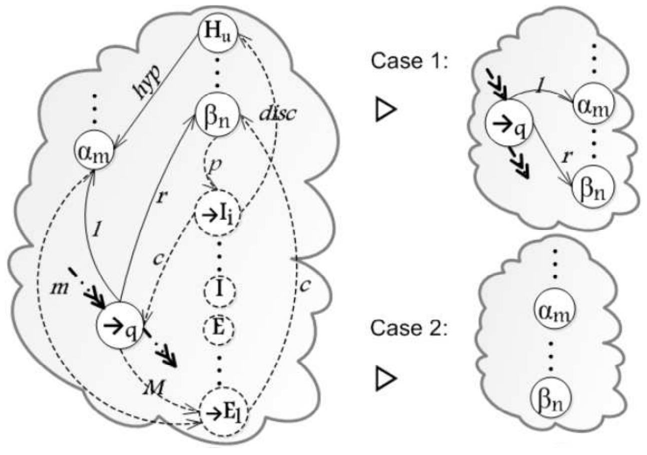


Figure 5.12: Elimination of ➔I followed by ➔E: Cases 1 and 2.

i. *If the D-node* H_u*, discharged by* $\rightarrow\mathrm{I}_i$*, has* n *outgoing edges with label hyp then repeat* n*-times edges in the minor sub-sequence of* $\rightarrow\mathrm{E}_l$*.*

ii. *Remove R-nodes* $\rightarrow\mathrm{I}_i$*,* $\rightarrow\mathrm{E}_l$ *and* H_u*, and their edges.*

iii. *Remove R-nodes* $\rightarrow\mathrm{I}_i$ *and* $\rightarrow\mathrm{E}_l$*, and their edges.*

iv. *If the F-node* $\wedge_q$ *only has outgoing edges to sub-formulas then remove it (see case 2 in Figure 5.12).*

(b) *Else If* $\rightarrow\mathrm{I}_i$ *represents two R-nodes then (see case 3 in Figure 5.13):*

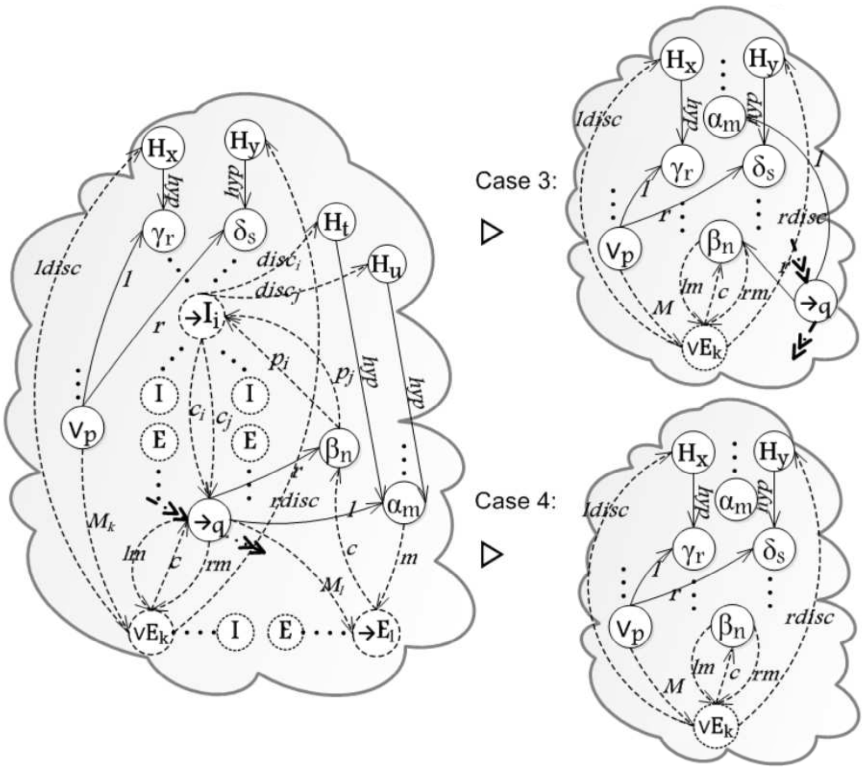


Figure 5.13: Elimination of $\rightarrow$I followed by $\rightarrow$E: Cases 3 and 4.

i. *Remove R-nodes* $\rightarrow\mathrm{I}_i$*,* $\rightarrow\mathrm{E}_l$*,* H_t *and* H_u*, and their edges.*

ii. *Eliminate edges:* $\rightarrow_q \xrightarrow{lm} \vee\mathrm{E}_k$*,* $\rightarrow_q \xrightarrow{rm} \vee\mathrm{E}_k$ *and* $\vee\mathrm{E}_k \xrightarrow{c} \rightarrow_q$*.*

iii. *If the F-node* $\rightarrow_q$ *only has outgoing edges to sub-formulas then remove it (see case 4 in Figure 5.13).*

iv. *Add edges:* $\beta_n \xrightarrow{lm} \vee\mathrm{E}_k$*,* $\beta_n \xrightarrow{rm} \vee\mathrm{E}_k$ *and* $\vee\mathrm{E}_k \xrightarrow{c} \beta_n$*.*

v. *Incorporate the R-node sequence with conclusion* α_m*, as defined in Definition 25, in left and right minor sub-sequences of* $\vee\mathrm{E}_k$*.*

(c) *Else (see case 5 in Figure 5.14)*

i. *Remove R-nodes* $\rightarrow\mathrm{I}_i$ *and* H_t*, and their edges.*

ii. *Eliminate edges:* $\rightarrow_q \xrightarrow{lm} \vee\mathrm{E}_k$, $\rightarrow_q \xrightarrow{rm} \vee\mathrm{E}_k$ *and* $\vee\mathrm{E}_k \xrightarrow{c} \rightarrow_q$.

iii. *Add edges:* $\beta_n \xrightarrow{lm} \vee\mathrm{E}_k$, $\beta_n \xrightarrow{rm} \vee\mathrm{E}_k$ *and* $\vee\mathrm{E}_k \xrightarrow{c} \beta_n$.

iv. *Incorporate the node* $\rightarrow\mathrm{E}_l$*, as defined in Definition 25, in the right minor subsequence of* $\vee\mathrm{E}_k$ *as last inference and the R-node sequence with conclusion* α_m *in the left minor subsequence.*

2. *Otherwise eliminate* maximal formulas *between R-nodes* $\rightarrow\mathrm{I}_i$ *and* $\rightarrow\mathrm{E}_l$*.*

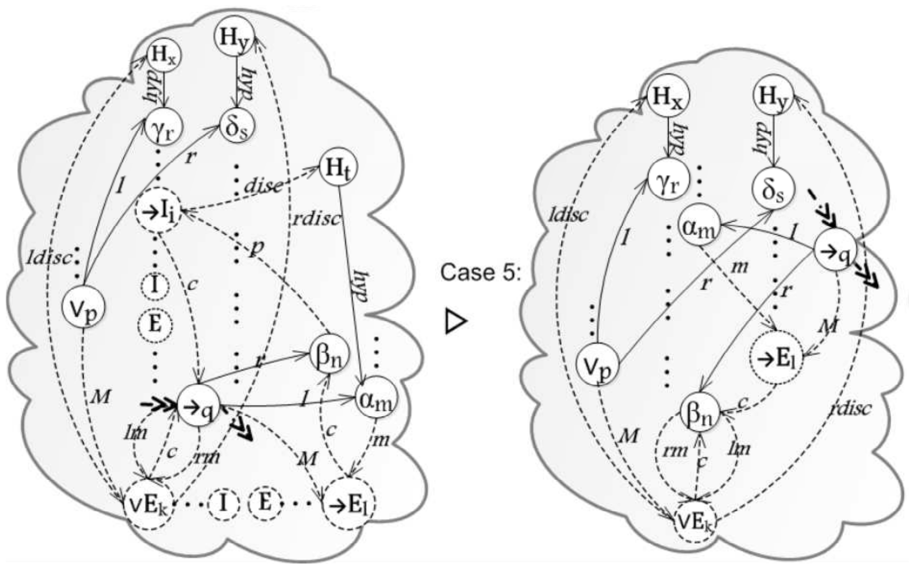


Figure 5.14: Elimination of $\rightarrow$I followed by $\rightarrow$E: Case 5.

Definition 27 *(1) For* $n_i \in V$*, a* p-path *in propositional mimp-graph is a sequence of vertices and edges of the form:* $n_1 \xrightarrow{lbl_1} n_2 \xrightarrow{lbl_2} \ldots \xrightarrow{lbl_{k-2}} n_{k-1} \xrightarrow{lbl_{k-1}} n_k$*, such that* n_1 *is a hypothesis formula node,* n_k *is the conclusion formula node,* n_i *alternating between a rule node and a formula node. Edges* lbl_i *alternate between two types of edges: the first is* $lbl_j \in \{rm, lm, m, M, rp, lp, p\}$ *and the second* $lbl_j = c$*. (2) A* branch *in propositional mimp-graph is an initial part of a* p-path *which stops at the conclusion F-node of the graph or at the first minor (or left) premise whose major (or right) premise is the conclusion of a rule node.*

Lemma 6 *If* G *is a propositional mimp-graph with a maximal formula* m *and* G' *is obtained from* G *by eliminating* m*, then* G' *is also a propositional mimp-graph.*

Proof: We use Lemma 5. All nodes in G' are of the right form: P, K, E, I, $\mathrm{E}^{\vee}$, $\mathrm{I}^{\vee}$, $\mathrm{E}^{\wedge}$, $\mathrm{I}^{\wedge}$, H or C. We verify that G' has one ingoing edge with label *conc* to the

D-node with label C and that is acyclic and connected. Finally, an inferential order on G' (as defined in Definition 13) between rule nodes must preserve. ■

5.2.2 Normalization proof

Just as with mimp-graphs, we shall also construct the normalization proof for these extended mimp-graphs. This proof is guided by the normalization measure. That is, the general mechanism from the proof determines that a given mimp-graph G should be transformed into a non-redundant mimp-graph by applying of reduction steps and at each reduction step the measure must be decreased. The normalization measure will be the number of maximal formulas in the mimp-graph.

Theorem 5 (Normalization) *Every propositional mimp-graph G can be reduced to a normal propositional mimp-graph G' having the same hypotheses and conclusion as G. Moreover, for any standard tree-like natural deduction Π, if $G := G_\Pi$ (the F-minimal mimp-like representation of Π, cf. Theorem 3), then the size of G' does not exceed the size of G, and hence also Π.*

Remark 2 *The second assertion sharply contrasts to the well-known exponential speed-up of standard normalization. Note that the latter is a consequence of the tree-like structure of standard deductions having different occurrences of equal hypotheses formulas, whereas all formulas occurring in F-minimal mimp-like representations are pairwise distinct.*

Proof: This characteristic of preservation of premises and conclusions of the derivation is proved naturally. Through an inspection of each elimination of maximal formula is observed that the reduction step (see Definition 26) of the propositional mimp-graph does not change the set of premises and conclusions (indicated by D-nodes H and C) of the derivation that is being reduced.

In addition, the demonstration of this theorem has two primary requirements. First, we guarantee that through the elimination of maximal formulas in the propositional mimp-graph, cannot generate more maximal formulas.The second requirement is to guarantee that during the normalization process, the normalization measure adopted is always reduced.

The first requirement is easily verifiable through an inspection of each case in the elimination of maximal formulas. Thus, it is observed that no case produces more maximal formulas. The second requirement is established through the normalization procedure (see Section 5.2.2) and demonstrated through an analysis of existing cases in the elimination of maximal formulas in mimp-graphs. To support this statement, it is used the notion of normalization measure, we adopt as measure of complexity (induction parameter) the number of maximal formulas $Nmax(G)$. Besides, as already mentioned, working with F-mimimal mimp-graph representations we can use as optional inductive parameter the ordinary size of mimp-graphs. ■

Normalization Process

We know that a specific propositional mimp-graph G can have one or more maximal formulas represented by $M_1, ..., M_n$. Thus, the normalization procedure is:

1. Choose a maximal formula represented by M_k.
2. Identify the respective number of maximal formulas $Nmax(G)$.
3. Eliminate M_k as defined in Definition 26, creating a new graph G.
4. In this application one, of the following six cases may occur:
   a) The maximal formula is removed (case 1 in all eliminations of maximal formulas).
   b) The maximal formula is removed but the formula node is maintained, and, $Nmax(G)$ is decreased (case 2 in all eliminations of maximal formulas);
   c) Two maximal formula are removed (case 3 in all eliminations of maximal formulas).
   d) Two maximal formula are removed but the formula node is maintained, hence $Nmax(G)$ is decreased (case 4 in all eliminations of maximal formulas).
   e) The maximal formula is removed, the formula node is maintained and R-node sequence reordered, hence $Nmax(G)$ is decreased (case 5 in all eliminations of maximal formulas).
   f) All maximal formulas are removed.
5. Repeat this process until the normalization measure $Nmax(G)$ is reduced to 0 and G becomes *a normal propositional mimp-graph.*

Since the process of the eliminating a maximal formula on propositional mimp-graphs always ends in the elimination of at least one maximal formula, and with the decrease in the number of vertices of the graph, we can say that this normalization theorem is directly a *strong normalization theorem.*

5.3 Proof-graphs for first order logic

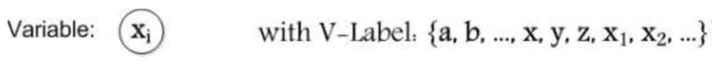
In this section we extend mimp-graphs for propositional logic defined in Section 5.1 to first order logic. This extension can be carried through without much ado and thus we show the robustness of the concept of mimp-graphs.

In Section 5.3.1 we give the definition of these mimp-graphs for first order logic, called *mimp-fol*, starting from definitions for terms, formula graphs of first order logic, and rule nodes for mimp-fol, then the Section 5.3.2 show two examples of mimp-fol. The set of transformations for normalization in mimp-fol is given in Section 5.3.3.

5.3.1 Definition

According to the language of first order logic for Gentzen-Prawitz style natural deduction (Gentzen 1969) (Prawitz 2006) introduced in Chapter 2, we extend mimp-graphs (defined in Chapter 3) to first order logic as follows.

- Variables are represented by nodes as follows:



  It is not necessary to differentiate between free and bound variables, since this will be made explicit by label in edges of the graph.

- Formulas are represented by formula graphs that are composed by formula nodes; thus the definition of formula graphs has three added inductive graphs:

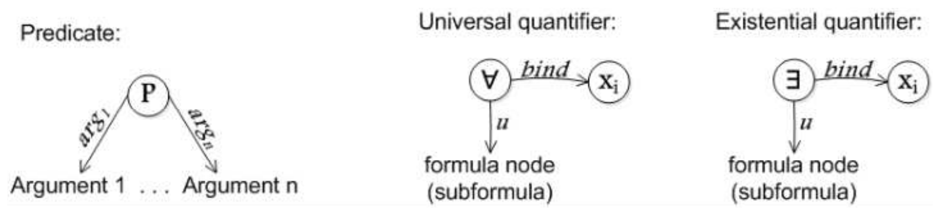


Figure 5.15 shows how bound variables appear in formula graphs; they are to be shared as much as possible. Like hypothesis (linked to D-node with label H) may only be discharged once, variables may not be bound by more than one quantifier. Also, quantifiers may not bind variables outside their direct subformula, their scope.

In mimp-fol, rule nodes operate conveniently on the root node (primary connective) of a formula-graph. Since quantifier rules of mimp-fol affect variables and we are sharing subformulas and since variables are different before and after substitution, we cannot share one formula graph as the "before and

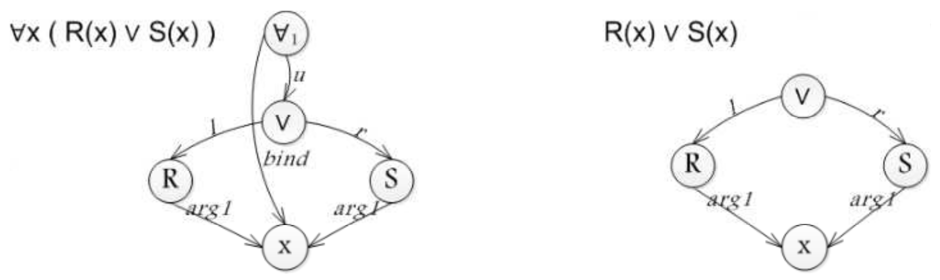


Figure 5.15: Bound and free variables in formula graphs.

after" of a substitution. Instead, two subgraphs are required that are different in the variable that is substituted for. The substitution explicit by an edge labeled *subs.by* (see Definition 28). In the following example we can identify two formula graphs as the premise and conclusion of a substitution, P and $P[x/t]$.

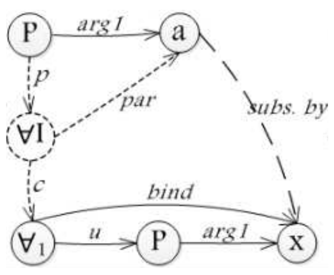


Figure 5.16: Premise and conclusion of a substitution.

Now we have the set of rules added in the extension:

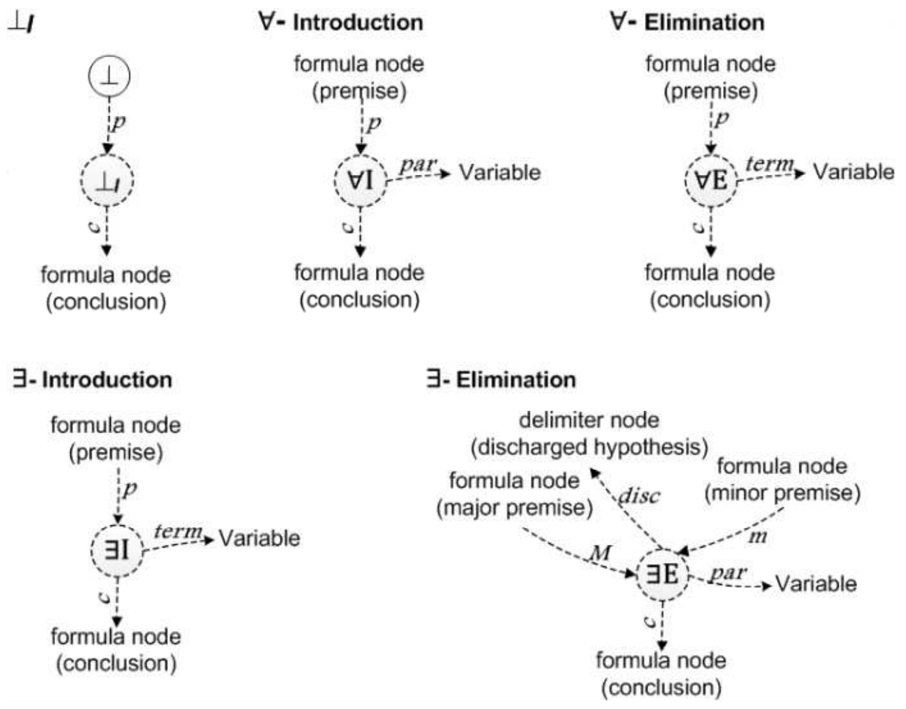
Definition 28 *A* mimp-fol *G is a directed graph* ⟨V, E, L⟩ *where:* V *is a set of nodes,* L *is a set of labels,* E *is a set of* edges ⟨$v \in$ V, $t \in$ L, $v' \in$ V⟩, *where v is the source and v' the target.*

The mimp-fol is defined recursively as follows:

pmimp *Every construction rule for propositional mimp-graphs (Definition 23) is a construction rule for mimp-fol.*

∀I *If G_1 is a mimp-fol, containing a node: α_m linked to the D-node C, then the graph G is defined as G_1 with*

1. *the removal of the ingoing edge in the node C;*
2. *an R-node* $\forall \mathrm{I}_i$ *at the top position;*
3. *duplicating: the graph of α_m with the substitution of x for t ($\alpha_m[x/a]$);*
4. *an F-node* $\forall_t$*;*



⊥I
formula node (conclusion)
∀- Introduction
formula node (premise)
Variable
formula node (conclusion)
∀- Elimination
formula node (premise)
Variable
formula node (conclusion)
∃- Introduction
formula node (premise)
Variable
formula node (conclusion)
∃- Elimination
delimiter node (discharged hypothesis)
formula node (major premise)
formula node (minor premise)
Variable
formula node (conclusion)

5. *edges:* $\alpha_m \xrightarrow{p_{new}} \forall\text{I}_i$, $\forall\text{I}_i \xrightarrow{c_{new}} \forall_t$, $\forall\text{I}_i \xrightarrow{par_{new}} t$, $\forall_t \xrightarrow{bind} x$, $\forall_t \xrightarrow{u} \alpha_m[x/a]$, $a \xrightarrow{subs.\ by} x$ *and* $\forall_t \xrightarrow{conc} C$;

is a mimp-fol under the proviso that 'a' does not occur in any variable node of the branch (see Figure 5.17).

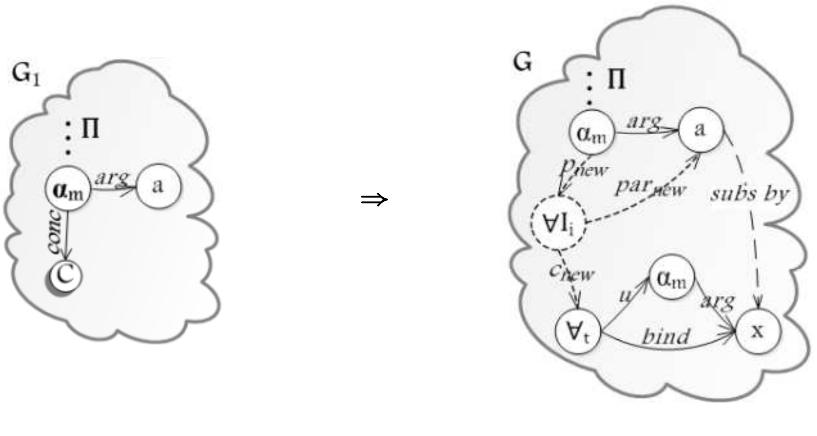


Figure 5.17: ∀-Introduction Rule.

∀E *If* G_1 *is a mimp-fol and contains the edge* $\forall_t \xrightarrow{u} \alpha_m$ *and the node* $\forall_t$ *linked to the delimiter node* C *then the graph* G *is defined as* G_1 *with*

1. *the removal of the ingoing edge in the node* C.
2. *an R-node* $\forall\text{E}_i$ *at the top position;*
3. *duplicating: the graph of* α_m *with the substitution of* x *for* t*–*$(\alpha_m[t/x])$*;*

4. *edges:* $\forall_t \xrightarrow{p_{new}} \forall\mathrm{E}_i$, $\forall\mathrm{E}_i \xrightarrow{c_i} \alpha_m[t/x]$, $\forall\mathrm{E}_i \xrightarrow{term_{new}} t$, $x \xrightarrow{subs.\ by} t$ *and* $\alpha_m[t/x] \xrightarrow{conc} C$,

is a mimp-fol (see Figure 5.18).

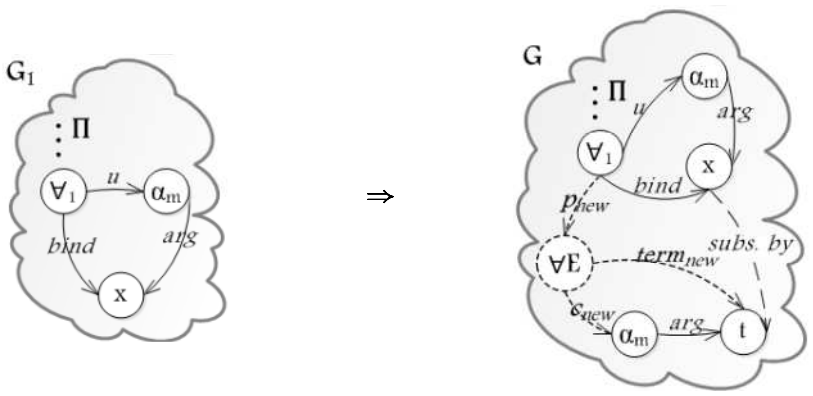


Figure 5.18: ∀-Elimination Rule.

∃I *If G_1 is a mimp-fol and contains the node α_m linked to the D-node C then the graph G is defined as G_1 with*

1. *the removal of the ingoing edge in the node C.*
2. *an R-node $\exists\mathrm{I}_i$ at the top position;*
3. *duplicating: the graph of α_m with the substitution of t for x $(\alpha_m[x/t])$;*
4. *edges:* $\alpha_m \xrightarrow{p_{new}} \exists\mathrm{I}_i$, $\exists\mathrm{I}_i \xrightarrow{c_{new}} \exists_t$, $\exists\mathrm{I}_i \xrightarrow{term_{new}} t$, $\exists_t \xrightarrow{bind} x$, $\exists_t \xrightarrow{u} \alpha_m[x/t]$, $t \xrightarrow{subs.\ by} x$ *and* $\exists_t \xrightarrow{conc} C$,

is a mimp-fol (see Figure 5.19).

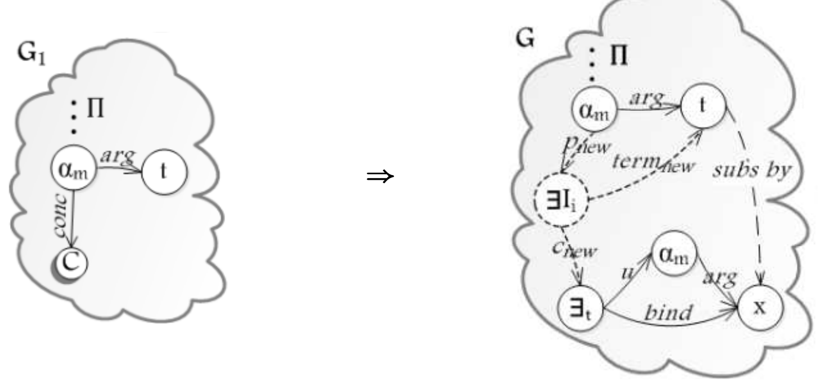


Figure 5.19: ∃-Introduction Rule.

∃E *If G_1 and G_2 are mimp-fol, and the graph obtained by $G_1 \oplus G_2$ (intermediate step) contains nodes: $\exists_t$ and σ_r linked to the D-node C; and α_m is linked to $\exists_t$ by $\exists_t \xrightarrow{u} \alpha_m$ and α_m is linked to D-node H_u, then the graph G is defined as $G_1 \oplus G_2$ with*

1. *the removal of the ingoing edges in the node C which were generated in the intermediate step ($G_1 \oplus G_2$);*
2. *an R-node $\exists\mathrm{E}_i$ at the top position;*
3. *edges: $\sigma_r \xrightarrow{m_{new}} \exists\mathrm{E}_i$, $\exists_t \xrightarrow{M_{new}} \exists\mathrm{E}_i$, $\exists\mathrm{E}_i \xrightarrow{par_{new}} a$, $\exists\mathrm{E}_i \xrightarrow{c_{new}} \sigma_r$, $\exists\mathrm{E}_i \xrightarrow{disc_{new}} H_u$, $x \xrightarrow{subs.\ by} a$ and $\sigma_r \xrightarrow{conc} C$;*

is a mimp-fol (see Figure 5.20).

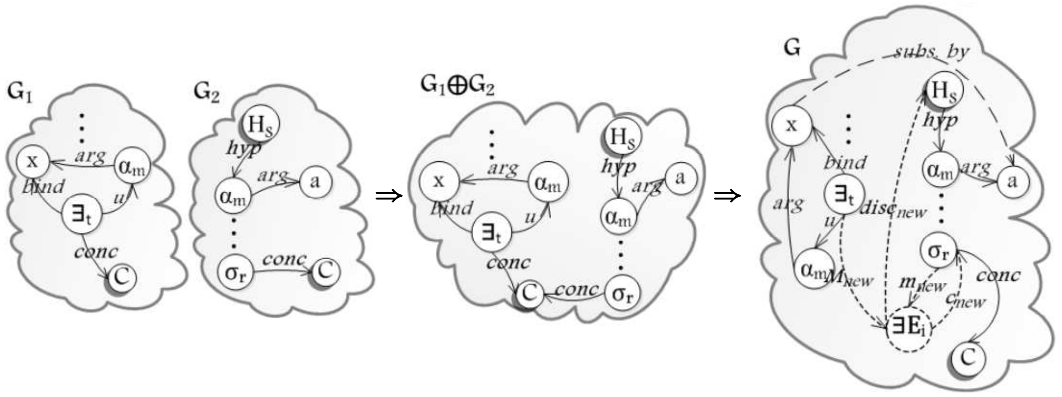


Figure 5.20: ∃-Elimination Rule.

⊥I *If G_1 is a mimp-fol and contains the node $\perp$ linked to the D-node C then the graph G is defined as G_1 with*

1. *the removal of the ingoing edge in the node C.*
2. *an R-node $\perp I_i$ at the top position;*
3. *edges: $\perp I_i \xrightarrow{c_{new}} \alpha_m$, $\perp \xrightarrow{p_{new}} \perp I_i$ and $\alpha_m \xrightarrow{conc} C$,*

is a mimp-fol.

5.3.2 Examples

By means of these examples we shall first of all show how our graphs represent deductions. Figure 5.21 shows a small proof with quantifiers, where the *term*-edge indicates the replaced term t in the inference scheme $\exists$I that has the conclusion $\exists y P(y)$ and the premise $P(y)[t/y] = P(t)$. The substitution is indicated by means of the *subs*-edge.

$$\dfrac{\exists x P(x) \qquad \dfrac{\dfrac{\dfrac{[P(t)]^1}{\exists y P(y)}\,\exists\mathrm{I} \qquad B}{\exists y P(y) \wedge B}\,\wedge\mathrm{I}}{\exists y P(y)}\,\wedge\mathrm{E}}{\exists y P(y)}\,\exists\mathrm{E},1 \qquad \Rightarrow$$

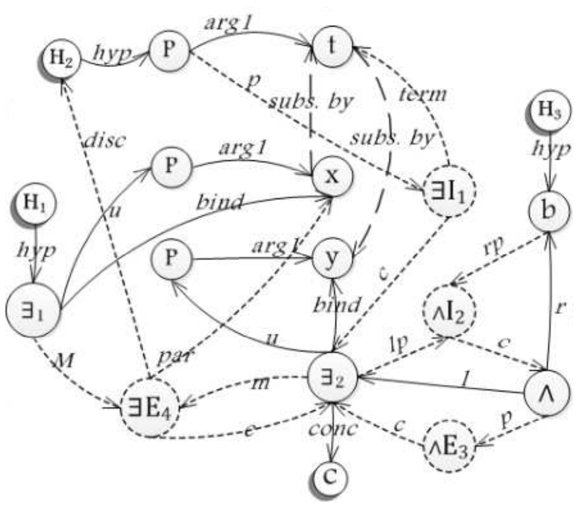


Figure 5.21: Example in proof-graphs for FOL

$$\dfrac{[\exists x \forall y F(x,y)]^1 \qquad \dfrac{\dfrac{\dfrac{[\forall y F(t_1,y)]^2}{F(t_1,t_2)}\,\forall\mathrm{E}}{\exists x F(x_1,t_2)}\,\exists\mathrm{I}}{\forall y_1 \exists x_1 F(x_1,y_1)}\,\forall\mathrm{I}}{\dfrac{\forall y_1 \exists x_1 F(x_1,y_1)}{\exists x \forall y F(x,y) \rightarrow \forall y_1 \exists x_1 F(x_1,y_1)}\,\rightarrow\mathrm{I}_2}\,\exists\mathrm{E}_1$$

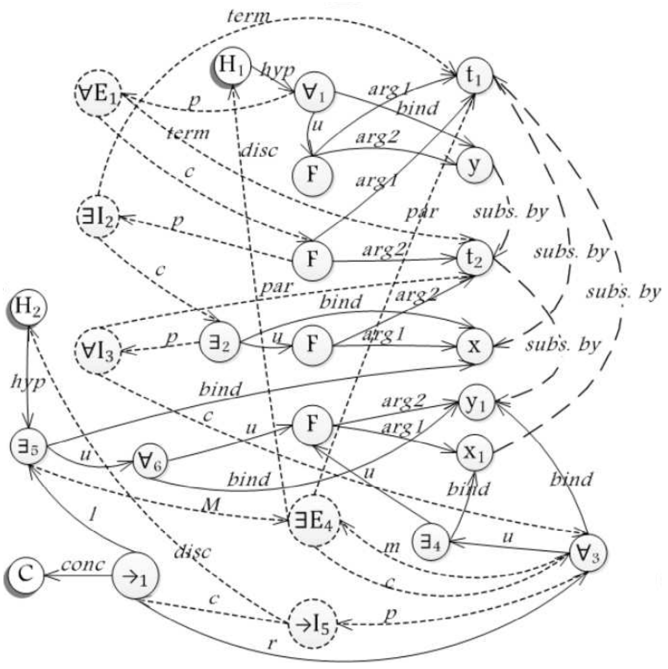
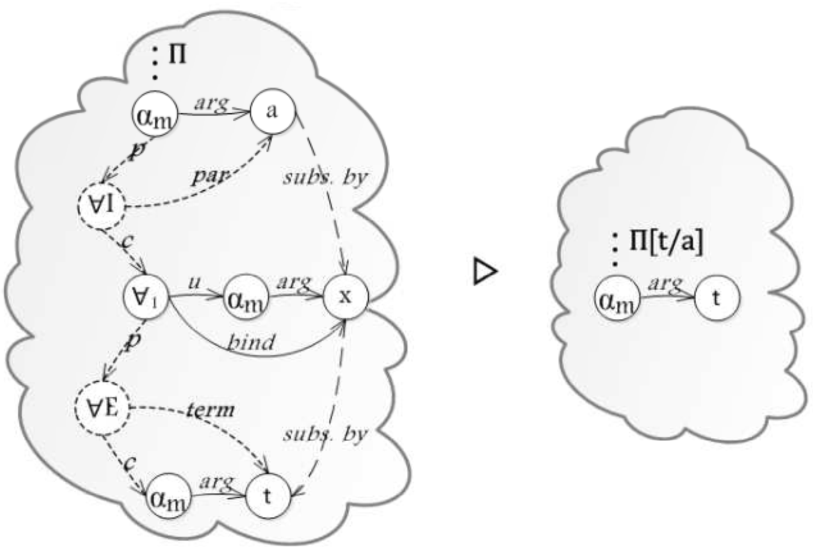


Figure 5.22: Example in mimp-graphs for FOL

5.3.3 Reductions for mimp-fol

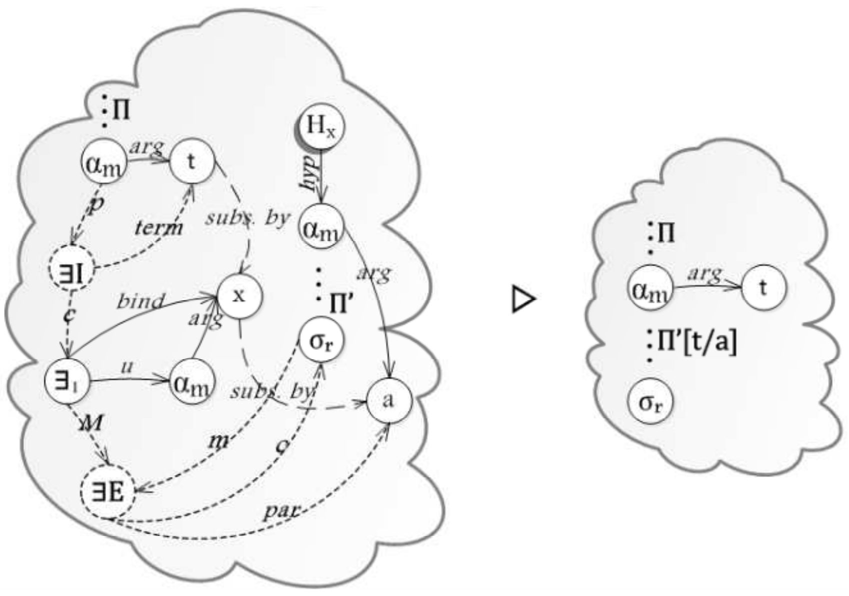
To obtain a normal derivation from any deduction, we transform it step by step, until no elimination rule is below an introduction rule. This process is called normalization, thus we give a set of transformations for first order logic proofs which preserve the information content of the original proof. We emphasize that for previous schemes of reduction (propositional logic), conclusions of maximal formulas remain shared, but now we have reduction schemes with added conclusions, we can compare it to the reduction scheme for natural deduction in Definition 3. Note that $\Pi[t/a]$ represents the resulting graph of replacing the label a on variable nodes by the label t.

Elimination of $\forall$I followed by $\forall$E In this reduction step are only preserved nodes and edges in the graph represented by Π, the formula graph α_m, the edge $\alpha_m \xrightarrow{arg} a$, the variable node with label a and the remaining graph represented by a cloud, then this label a on variable nodes is replaced by $t(\Pi[t/a])$.



Elimination of $\exists$I followed by $\exists$E Now, we preserve the graph represented by Π, the formula graph α_m, the edge $\alpha_m \xrightarrow{arg} t$, the variable node with label t, the graph Π', the formula graph σ_r, then the label a on variable nodes is replaced by t $(\Pi'[t/a])$.

then this label a on variable nodes is replaced by t.


Π
αm
arg
t
p
term
subs. by
∃I
c
bind
x
arg
∃1
u
αm
subs. by
M
m
c
par
∃E
Hx
hyp
αm
arg
Π'
σr
a
▷
Π
αm
arg
t
Π'[t/a]
σr

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

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

where f and t are the units *false* and *true*, $[S,...,S]$ is a *disjuntion* and $(S,...,S)$ is a *conjunction*. Atoms are denoted by $a, b, ...$ 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 $\overline{[a,b,c]}$ and $(\bar{a},(\bar{b},\bar{c}))$ 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

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

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

Table 6.2: System SKS_g

6.1.2 Proof graphs for deep inference

Our proof-graphs introduced in Chapter 3, explore, basically, the sub-formula 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)*$\}$*,*

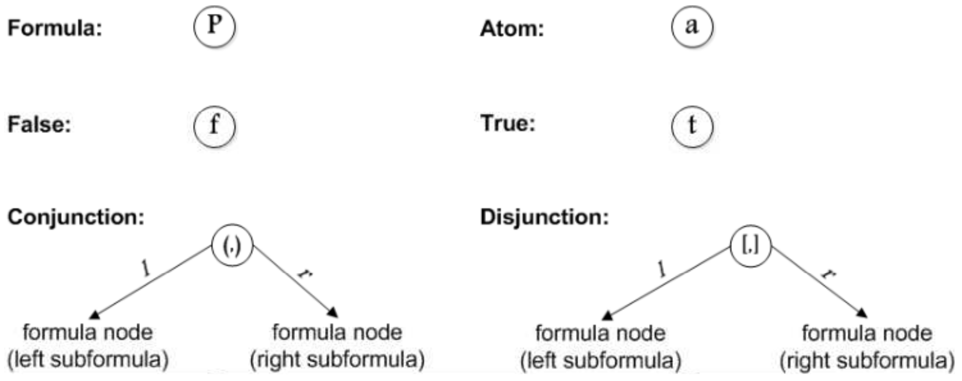

Formula:
P
Atom:
a
False:
f
True:
t
Conjunction:
(.)
l
r
formula node (left subformula)
formula node (right subformula)
Disjunction:
[,]
l
r
formula node (left subformula)
formula node (right subformula)

Figure 6.1: Formula nodes in Deep-graphs

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

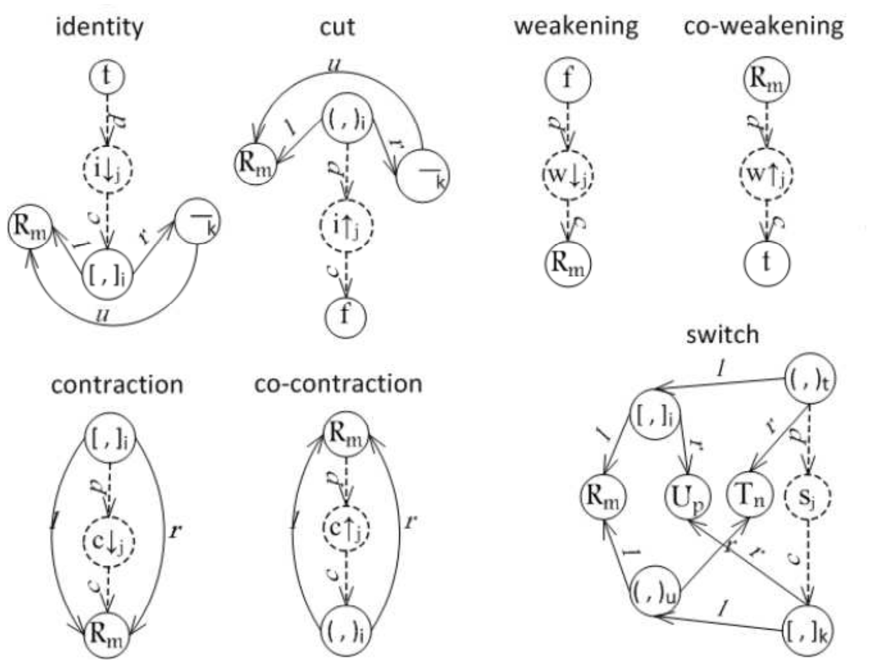
Deep-graphs are recursively defined as follows:

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*

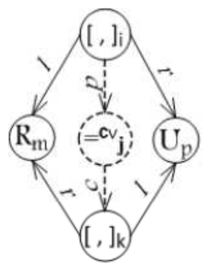
Structural Rules

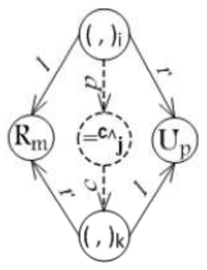
[1]We use the terms R_m, T_n and U_p to represent the principal connective of the formulas R, T and U, respectively.



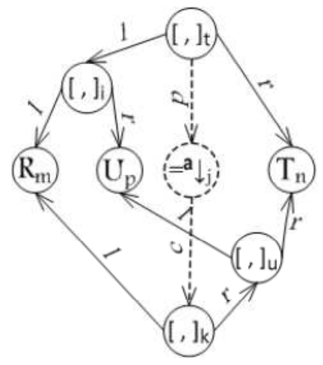
Logical Rules

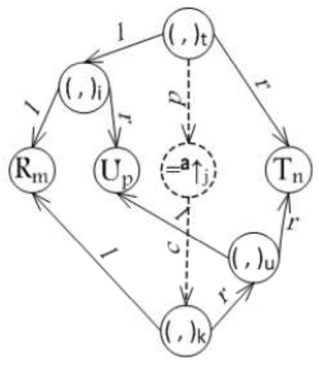
Commutativity



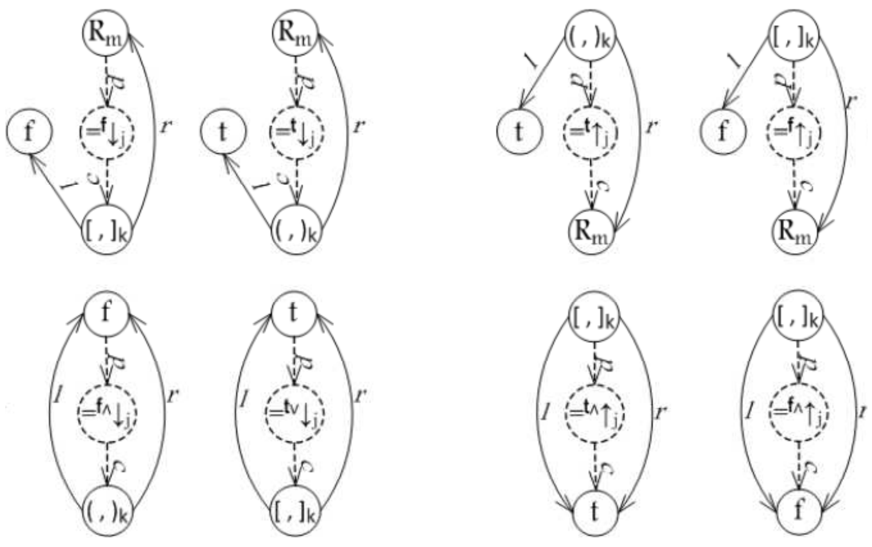


Associativity





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:

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

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

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

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

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

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;

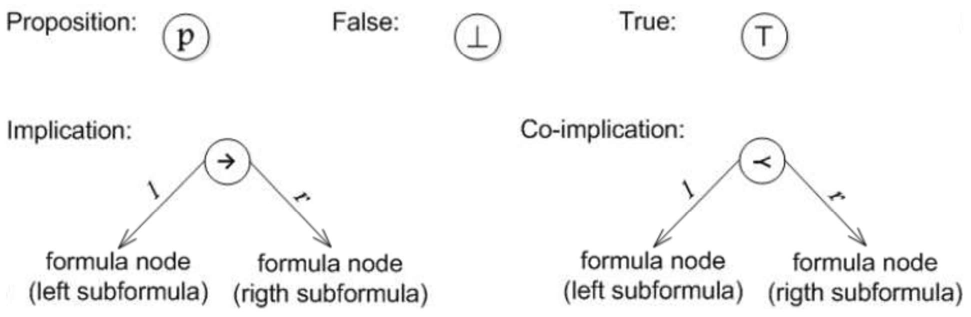
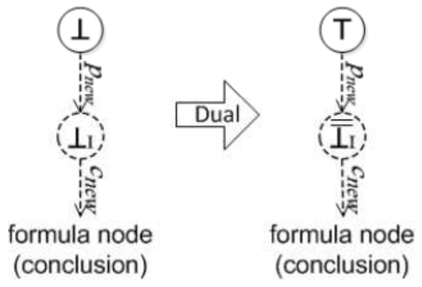

Proposition:
p
False:
⊥
True:
T
Implication:
→
l
r
formula node (left subformula)
formula node (rigth subformula)
Co-implication:
≺
l
r
formula node (left subformula)
formula node (rigth subformula)

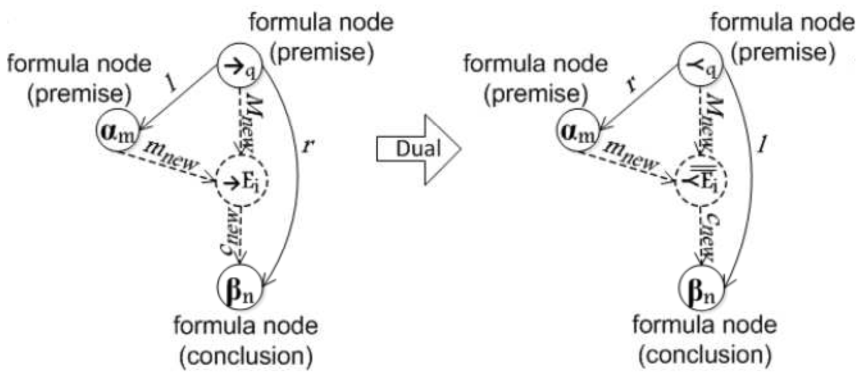
Figure 6.2: Formula nodes in 2Int-graph

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

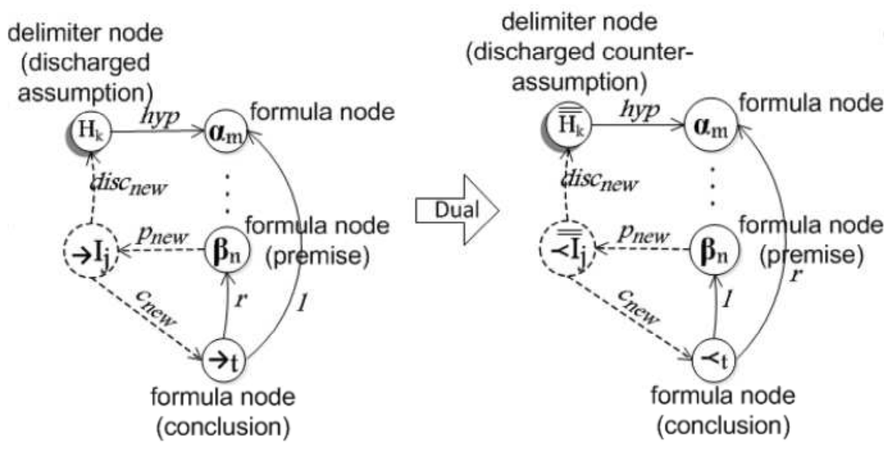
   Intuitionistic absurdity $\perp_I$ and its dual rule $\overline{\overline{\perp_I}}$.


⊥
p_{new}
$\perp_I$
c_{new}
formula node (conclusion)
Dual
T
p_{new}
$\overline{\overline{\perp_I}}$
c_{new}
formula node (conclusion)

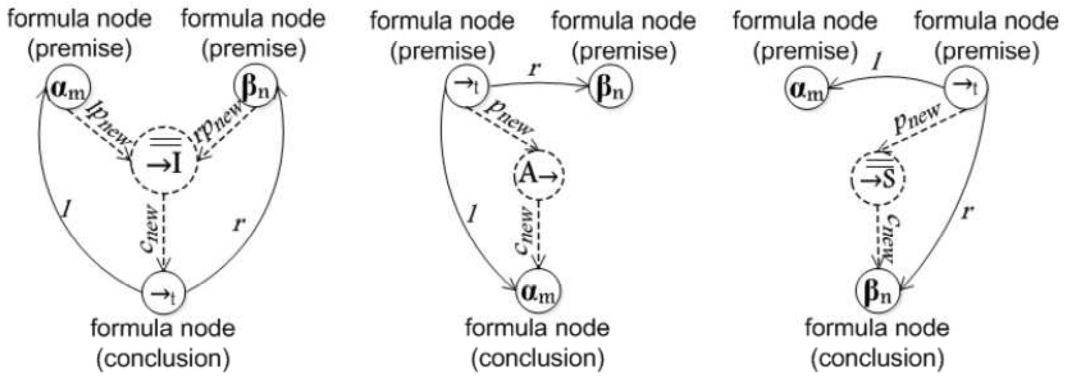
Implication Elimination →E and its dual rule $\overline{\overline{\prec E}}$



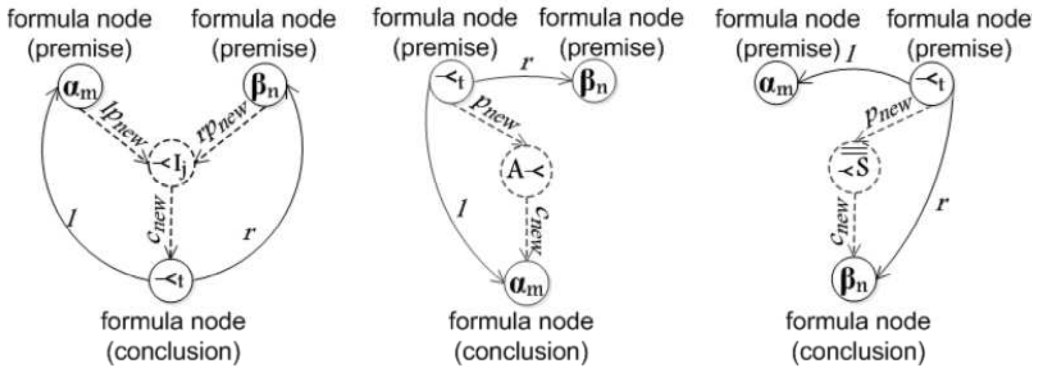
Implication Introduction →I and its dual $\overline{\overline{\prec I}}$



$\overline{\overline{I}}$, $A\rightarrow$ 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*.

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

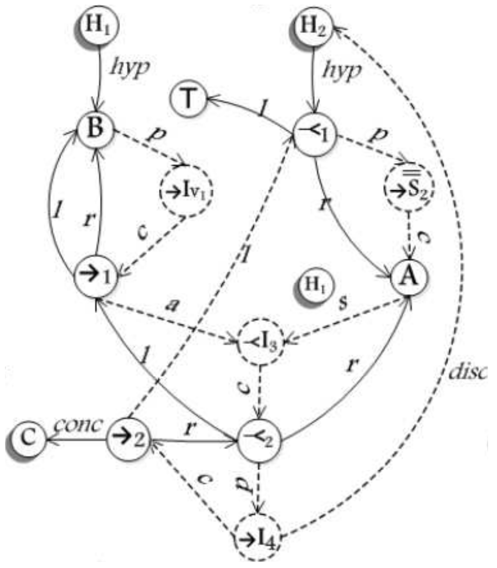


Figure 6.3: Translation of derivation in *N2Int* to *2Int-graph*

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

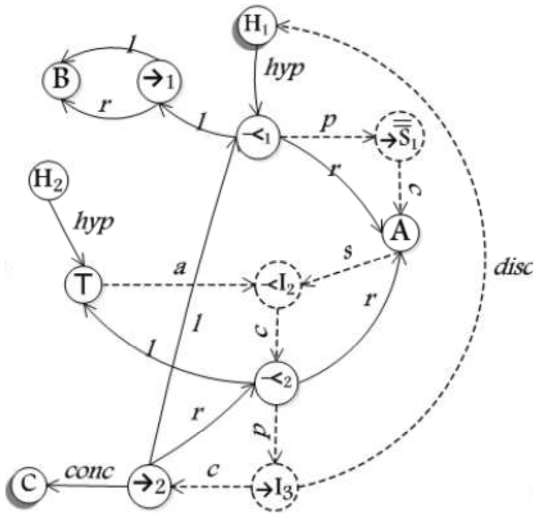


Figure 6.4: Translation of derivation in *N2Int* to *2Int-graph*

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.

7
Conclusions and future works

Our proof-graph representation, which we call mimp-graph, defined in Chapter 3, was introduced through definitions and examples, mainly devised for extracting proof-theoretic properties from proof system.

That is, we have tackled one of our research tracks. Mimp-graph preserves the ability to represent proofs in Natural Deduction and its minimal formula representation is a key feature of the mimp-graph structure, because as we saw earlier, it is easy to determine maximal formulas and upper bounds on the length of reduction sequences leading to normal proofs.

Thus a normalization theorem can be proved by counting the number of maximal formulas in the original derivation. The strong normalization is a direct consequence of such normalization, since any reduction decreases the corresponding measures of derivation complexity.

The results presented for mimp-graph are naturally extended for propositional mimp-graphs in Chapter 5. We also get strong normalization for normalization process that is proved by counting the number of maximal formulas in the original derivation.

Other merit of the present thesis is the treatment of sharing for inference rules, in addition to formula sharing, developed in Chapter 4, representing graphs more compactly, which is performed during the construction process of the graph. This feature is very important, since we intend to use this mimp-graph approach in automatic theorem provers.

In the construction process when similar formulas are found, the process of producing the unifier would consume linear time. We do not implement a process of searching for similar formulas in our graph but we estimate that the resources consumed in such searching would be compensated by the reduction of necessary resources to build the proof-graph.

Other contribution is the representation in graphs for first order logic. With our approach, we get a better view on the behaviour of variables inside a proof: variable binding in both quantifiers and inferences.

Our experimentation with other logics as deep inference has resulted in a representation for Calculus of Structures and that it preserves 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. Our proof-graph representation also has the ability to access sub-formulas, allowing the inference rules to be applied in any place deep inside a formula graph, just like deep inference. So, we get for our graph the same good characteristics of deep inference. We intend, as future work, to propose a normalization procedure for deep-graphs, where we will use the technique of reducing cuts similar to what is done in normalization for propositional mimp-graphs.

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.

Finally, we left the details of an efficient implementation of these graphs for future work. This is a preliminary step into investigating how a theorem prover based on graphs can be more efficient than usual theorem provers.

Bibliography

Alves, S., Fernández, M. & Mackie, I. (2011), A new graphical calculus of proofs, *in* R. Echahed, ed., 'Proceedings of TERMGRAPH 2011', Vol. 48 of *EPTCS*, pp. 69–84. DOI `10.4204/EPTCS.48.8`.

Bonet, M. L. & Buss, S. R. (1993), 'The deduction rule and linear and near-linear proof simulations', *Journal of Symbolic Logic* **58**(2), 688–709. DOI `10.2307/2275228`.

Brünnler, K. (2004), *Deep Inference and Symmetry in Classical Proofs*, Logos Verlag, Berlin. URL `http://www.iam.unibe.ch/~kai/Papers/phd.pdf`.

da Costa, V. G. (2007), Compactação de Provas Lógicas, PhD thesis, Departamento de Informática, PUC–Rio. URL `http://www.maxwell.lambda.ele.puc-rio.br/Busca_etds.php?strSecao=resultado&nrSeq=10018@2`.

Finger, M. (2005), Dag sequent proofs with a substitution rule, *in* 'We will show Them – Essays in honour of Dov Gabbay 60th birthday', Vol. 1 of *Kings College Publications*, Kings College, London, pp. 671–686. URL `http://www.ime.usp.br/~mfinger/home/papers/FW04.pdf`.

Gentzen, G. (1969), 'Investigations into logical deduction', pp. 68–213.

Geuvers, H. & Loeb, I. (2007), 'Natural deduction via graphs: Formal definition and computation rules', *Mathematical. Structures in Comp. Sci.* **17**(3), 485–526. DOI `10.1017/S0960129507006123`. ISSN 0960-1295.

Girard, J.-Y. (1996), Proof-nets: The parallel syntax for proof-theory, *in* 'Logic and Algebra', Dekker, pp. 97–124. URL `iml.univ-mrs.fr/~girard/Proofnets.ps.gz`.

Girard, J.-Y., Lafont, Y. & Regnier, L. (1995), *Advances in Linear Logic*, Cambridge University Press. Proceedings of the Workshop on Linear Logic, Ithaca, New York, June 1993.

Gordeev, L., Haeusler, E. H. & Costa, V. G. (2009), 'Proof compressions with circuit-structured substitutions', *Journal of Mathematical Sciences* **158**(5), 645–658. DOI `10.1007/s10958-009-9405-3`. ISSN 1573-8795.

Goré, R. (1998), 'Substructural logics on display', *Logic Journal of the IGPL* **6**(3), 451–504.

Goré, R. & Postniece, L. (2010), 'Combining derivations and refutations for cut-free completeness in bi-intuitionistic logic', *Journal of Logic and Computation* **20**(1), 233–260. DOI `10.1093/logcom/exn067`. URL `http://logcom.oxfordjournals.org/content/20/1/233.abstract`.

Goré, R., Postniece, L. & Tiu, A. (2010), Cut-elimination and proof search for bi-intuitionistic tense logic, *in* 'Advances in Modal Logic', pp. 156–177.

Guglielmi, A. (2007), 'A system of interaction and structure', *ACM Transactions on Computational Logic.* DOI `10.1145/1182613.1182614`. ISSN 1529-3785.

Guglielmi, A. & Gundersen, T. (2008), 'Normalisation control in deep inference via atomic flows', *Logical Methods in Computer Science.* DOI `10.2168/LMCS-4(1:9)2008`. URL `http://dx.doi.org/10.2168/LMCS-4(1:9)2008`.

Haeusler, E. (2013), 'A proof-theoretical discussion on the mechanization of propositional logics', *Electronic Proceedings in Theoretical Computer Science Vol. 113, pp. 7-8.* DOI `10.4204/EPTCS.113`. URL `http://rvg.web.cse.unsw.edu.au/eptcs/content.cgi?LSFA2012#EPTCS113.3`.

Haeusler, E. H. (in press), 'How many times do we need an assumption?', *arXiv.* preprint arXiv:submit/0969871.

Menezes, B. & Haeusler, E. (2006), *Teoria das Categorias Para Ciencia da Computaçao*, Bookman Companhia ED. URL `http://books.google.com.br/books?id=c60HPgAACAAJ`.

Oliveira, A. G. & Queiroz, R. J. (2003), Geometry of deduction via graphs of proofs, *in* R. J. Queiroz, ed., 'Logic for Concurrency and Synchronisation', Vol. 15 of *Trends in Logic*, Springer Netherlands, pp. 3–88.

Pinto, L. & Uustalu, T. (2009), Proof search and counter-model construction for bi-intuitionistic propositional logic with labelled sequents, *in* 'TABLEAUX', pp. 295–309.

Prawitz, D. (2006), *Natural Deduction*, Dover books on mathematics, Dover Publications, Incorporated. URL http://books.google.com.pe/books?id=IosnAAAACAAJ.

Quispe-Cruz, M., de Oliveira, A. G., de Queiroz, R. J. G. B. & de Paiva, V. (2014), 'Intuitionistic n-graphs', *Logic Journal of IGPL* **22**(2), 274–285. DOI 10.1093/jigpal/jzt033. URL http://jigpal.oxfordjournals.org/content/22/2/274.abstract.

Rauszer, C. (1974), 'A formalization of the propositional calculus of h-b logic', *Studia Logica* **33**(1), 23–34. DOI 10.1007/BF02120864. ISSN 0039-3215. URL http://dx.doi.org/10.1007/BF02120864.

Rauszer, C. (1977), 'Applications of kripke models to heyting-brouwer logic', *Studia Logica* **36**(1-2), 61–71. DOI 10.1007/BF02121115. ISSN 0039-3215. URL http://dx.doi.org/10.1007/BF02121115.

Restall, G. (1977), 'Extending intuitionistic logic with subtraction'. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.45.8231.

Statman, R. (1979), 'Intuitionistic propositional logic is polynomial-space complete', *Theoretical Computer Science* **9**, 67–72. DOI 10.1016/0304-3975(79)90006-9. ISSN 0304-3975. URL http://www.sciencedirect.com/science/article/pii/0304397579900069.

van Dalen, D. (1994), *Logic and structure (3. ed.)*, Universitext, Springer.

Wansing, H. (2013), 'Falsification, natural deduction and bi-intuitionistic logic', *Journal of Logic and Computation.* DOI 10.1093/logcom/ext035. URL http://logcom.oxfordjournals.org/content/early/2013/07/17/logcom.ext035.abstract.