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 $\rightarrow\mathrm{I}_2$, $\rightarrow\mathrm{E}_3$, $\rightarrow\mathrm{E}_4$ and $\rightarrow\mathrm{I}_6$, $\rightarrow\mathrm{E}_7$, $\rightarrow\mathrm{E}_8$ 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

$$
\begin{prooftree}
\AxiomC{$[p]^1$}
\AxiomC{$[p]^1$}
\AxiomC{$p \to (p \to q)$}
\RightLabel{$\to E_8$}
\BinaryInfC{$p \to q$}
\RightLabel{$\to E_7$}
\BinaryInfC{q}
\RightLabel{$\to I_6$}
\UnaryInfC{$\boxed{p \to q}$}
\AxiomC{$(p \to q) \to p$}
\RightLabel{$\to E_5$}
\BinaryInfC{p}
\AxiomC{$[p]^1$}
\AxiomC{$[p]^1$}
\AxiomC{$p \to (p \to r)$}
\RightLabel{$\to E_4$}
\BinaryInfC{$p \to r$}
\RightLabel{$\to E_3$}
\BinaryInfC{r}
\RightLabel{$\to I_2$}
\UnaryInfC{$\boxed{p \to r}$}
\RightLabel{$\to E_1$}
\BinaryInfC{r}
\end{prooftree}
$$

$$\Downarrow trans$$

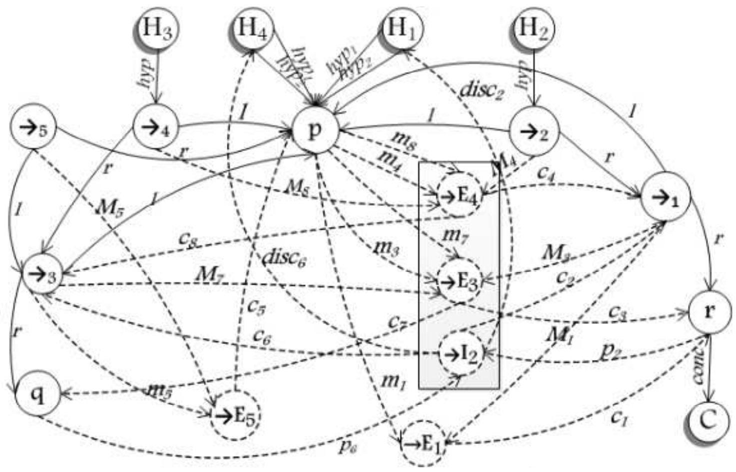
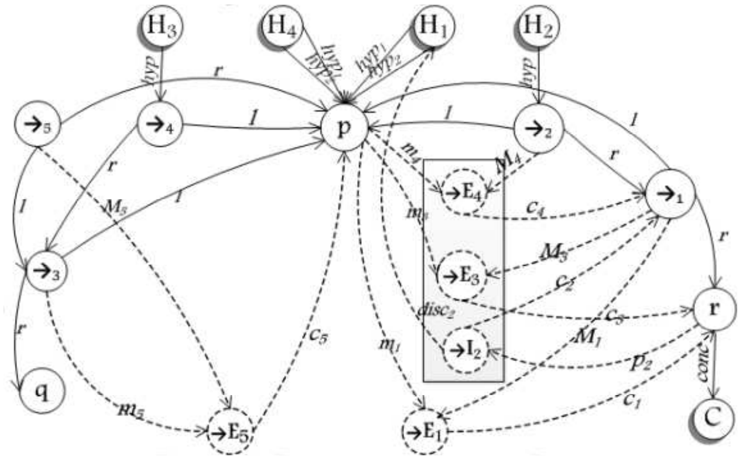


Figure 4.1: The transition from a natural deduction proof to a mimp-graph

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 $\to_3$ is the formula that attempts to reuse a derivation with a conclusion isomorphic to it and a premise isomorphic to $\to_4$, in the below illustration we see how it looks after sharing. Thus, the sequences of rules: $\to I_2$, $\to E_3$, $\to E_4$ and $\to I_6$, $\to E_7$, $\to E_8$, 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.



$\Downarrow$ *share*

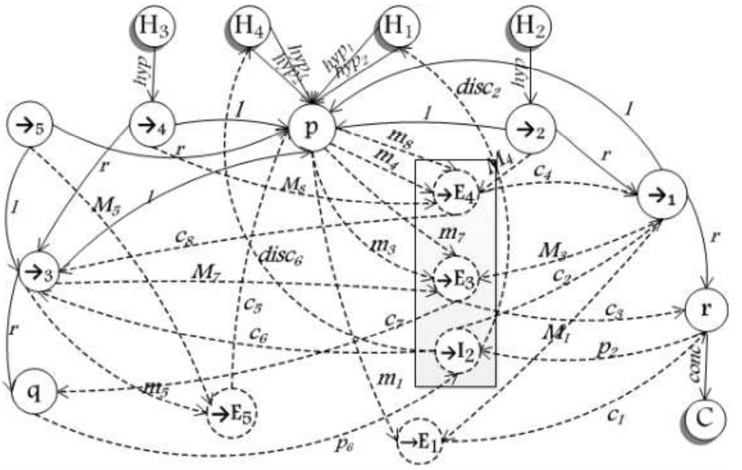


Figure 4.2: The transition before and after of sharing.

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 $\phi : V \to V'$ 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).

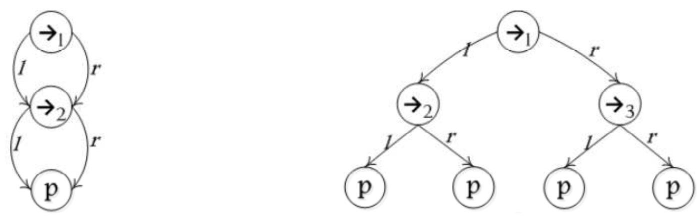


Figure 4.3: Formula $(p \to p) \to (p \to p)$ depicted as a formula graph (left side) and as an unfolding graph (right side).

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 σ_k; σ_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 α_k, where α_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 α_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:* $\forall i, j \in I$ $(\mathrm{Box}_i \cap \mathrm{Box}_j = \varnothing \vee \mathrm{Box}_i \subset \mathrm{Box}_j \vee \mathrm{Box}_j \subset \mathrm{Box}_i)$.

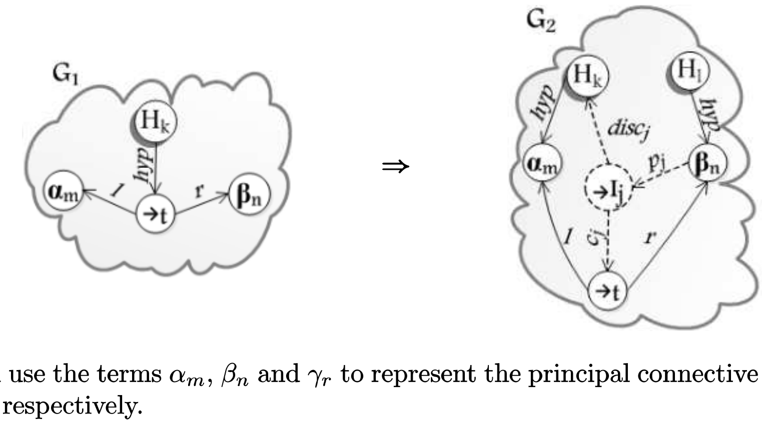
A smimp is defined recursively as follows:

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*).*

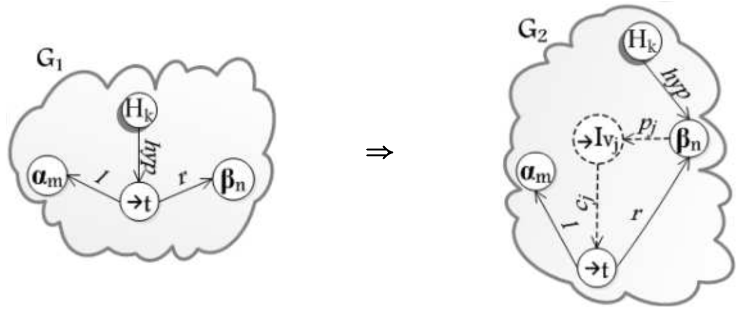


[1]We will use the terms α_m, β_n and γ_r to represent the principal connective of the formula α, β and γ respectively.

$\rightarrow$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 $\rightarrow\mathrm{Iv}_j \xrightarrow{c_j} \rightarrow_t$;*

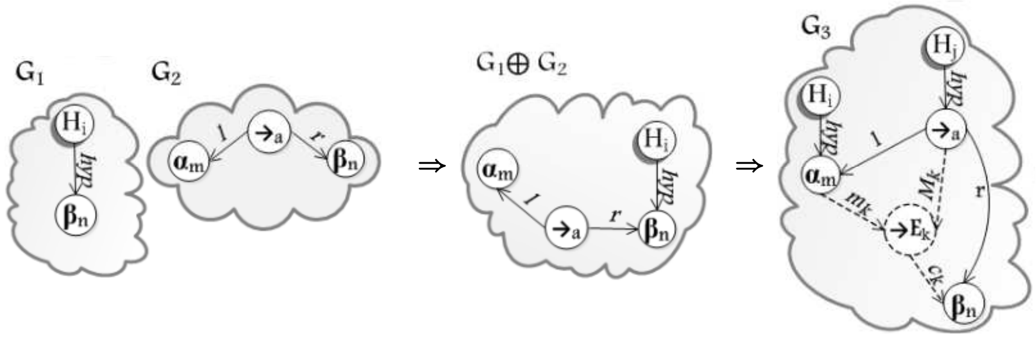
is a smimp (see figure below).



$\rightarrow$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 $\rightarrow\mathrm{E}_k \xrightarrow{c_k} \beta_n$;*

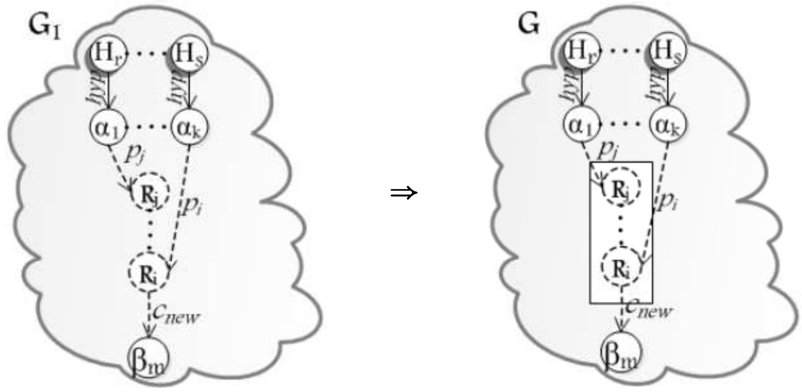
is a smimp (see figure below).



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).*

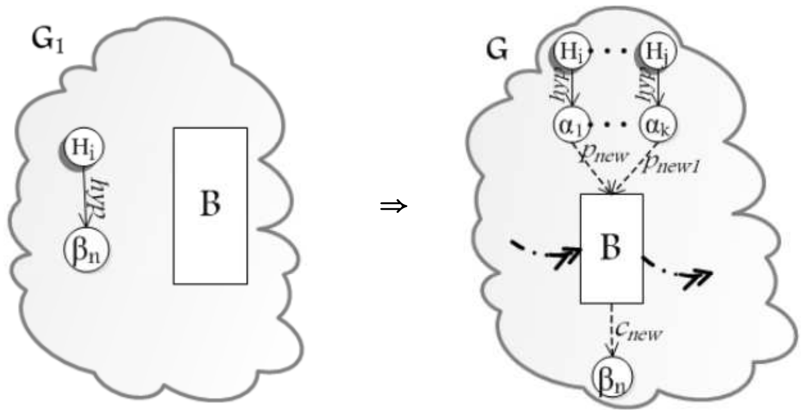


Share *If G_1 is a smimp containing the F-node β_n linked to a D-node H and there is a rule box* Box = $\{R_1, ..., R_m\}$ *and β_n is unifiable[3] with some conclusion graph of the box[4] and each element in $\alpha_1, ..., \alpha_k$ (desirable hypothesis) is unifiable with each premise[5] of the box, 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 $\to_n$ 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 $\to\mathrm{E}_k$ and has one (or n if the node is in a box) outgoing edges $\to\mathrm{E}_k \xrightarrow{c_k} \beta_n$, 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: $\alpha_m \xrightarrow{m_k} \to\mathrm{E}_k$ and $\to_q \xrightarrow{M_k} \to\mathrm{E}_k$, where α_m is a node type* **P** *or* **K**. *there are also more exactly two outgoing edges from the node $\to_a$: $\to_a \xrightarrow{l} \alpha_m$ and $\to_a \xrightarrow{r} \beta_n$.*

I *N has label $\to\mathrm{I}_j$ (or $\to\mathrm{Iv}_j$, if it discharges an hypothesis vacuously), has one (or n if the node is in a box) outgoing edge $\to\mathrm{I}_j \xrightarrow{c_j} \to_t$), and one (or zero for the vacuous case $\to\mathrm{Iv}$, or n if the node is in a box) outgoing edge ($\to\mathrm{I}_j$, $disc_j$, 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 $\to_t$: $\to_t \xrightarrow{l} \alpha_m$ and $\to_t \xrightarrow{r} \beta_n$ such that there is one (or zero for the case $\to\mathrm{Iv}$) 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 $h: M \to S_M$ 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 $M^{\to}$ 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 $\chi[X,Y] = (((X \to Y) \to X) \to X) \to Y$. 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}.

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

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

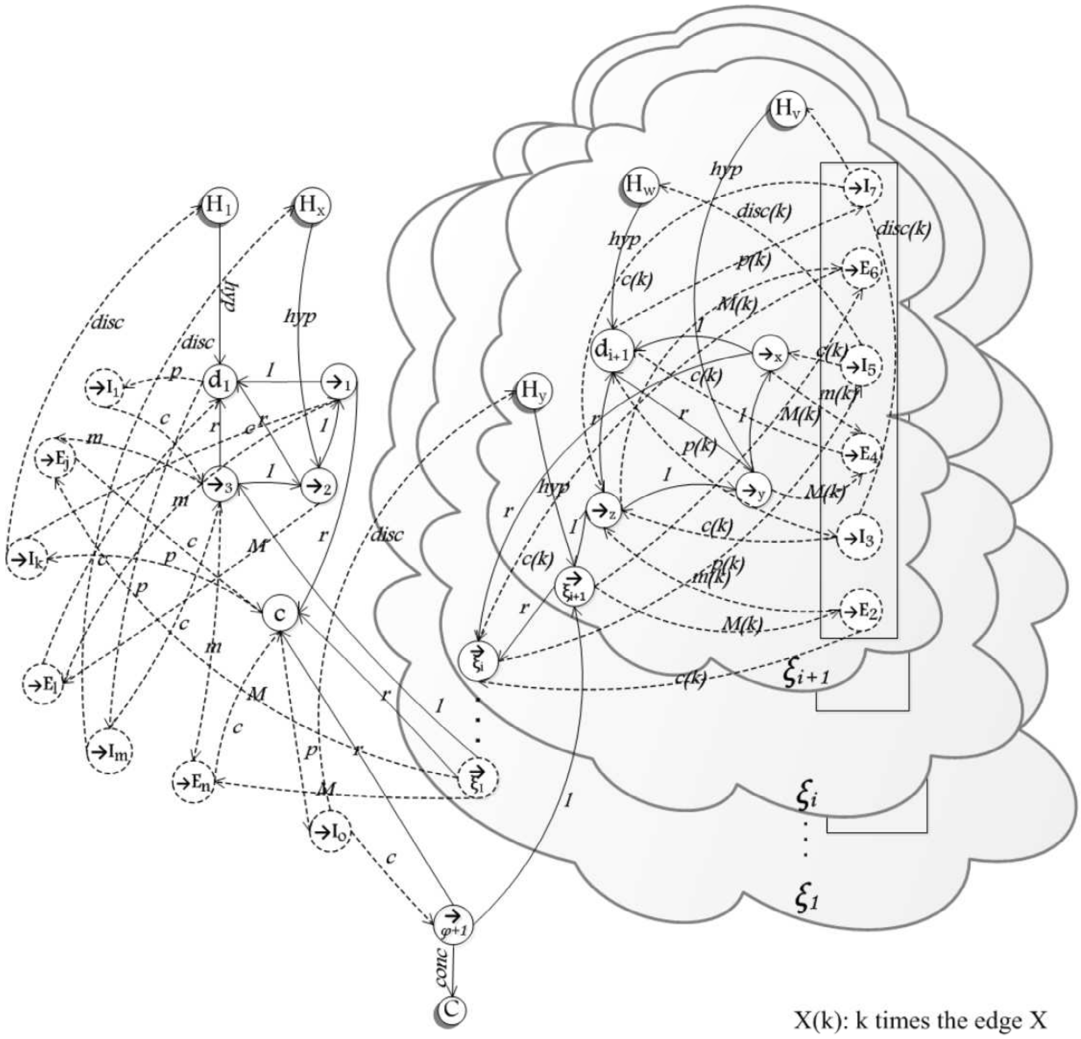


Figure 4.4: Smimp of φ_{k+1}

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.