Coherence for sharing proof-nets*

S. Guerrini[1], S. Martini[2], A. Masini[1]

[1] Dipartimento di Informatica, Università di Pisa,
Corso Italia, 40, I-56125 Pisa – Italy; `guerrini,masini@di.unipi.it`.
[2] Dipartimento di Matematica e Informatica, Università di Udine,
Via delle Scienze, 206, I-33100 Udine – Italy; `martini@dimi.uniud.it`.

ABSTRACT: Sharing graphs are a way of representing linear logic proof-nets in such a way that their reduction never duplicates a redex. In their usual presentations, they present a problem of coherence: if the proof-net N reduces by standard cut-elimination to N', then, by reducing the sharing graph of N we do *not* obtain the sharing graph of N'. We solve this problem by changing the way the information is coded into sharing graphs and introducing a new reduction rule (*absorption*). The rewriting system is confluent and terminating.

1 Introduction

Implementations of functional languages based on graph rewriting need sophisticated techniques to control the runtime duplication of subgraphs. From a theoretical point of view, we know after [Lév78] that given a normalizable λ-term there is an optimal (in the number of beta-reductions) reduction strategy to reach the normal form. Since, however, it is a parallel strategy (counting as a single step the simultaneous reduction of several redexes, those belonging to the same *family*), how to implement this strategy remained open until Lamping [Lam90] introduced his *sharing graphs*.

Sharing graphs are based on three main ideas. First, any time a duplication seems required (e.g., when a bound variable appears several times in the body of a term), it is not actually performed; it is instead indicated (in a somewhat lazy way) by specific (new) nodes in the graph (*fans*, in Lamping's terminology). Second, special reduction rules are added to perform the actual duplication in a controlled way (a redex will be never duplicated). Finally (and non trivially), there is a way to mark the boundary of the subgraph where duplication has to happen (again new nodes, the *brackets*). The reduction then proceeds in a distributed and asynchronous way, firing locally those reduction rules which apply. The crucial properties to show are then: (i) this asynchronous process terminates (if the term has a normal form); (ii) the normal form is (a possibly shared representative of) the normal form we would have reached we had done the reduction in the standard way; and (iii) no useless duplication is ever done (i.e., optimality of beta-reduction).

* Partially supported by: HCM Project CHRX-CT93-0046 and CNR GNSAGA (Guerrini,Martini); by BRA 8130 LOMAPS (Masini); by MURST 40% "Modelli della computazione e dei linguaggi di programmazione".

Following Lamping's breakthrough, several papers generalized and improved his result. First, Gonthier, Abadi, and Lévy [GAL92a, GAL92b] realized that Lamping's method was in fact a way to reduce linear logic proof-nets [Gir87] and that the information needed to mark the boundary of the subgraph to be duplicated, was a local and distributed representation of the (global) notion of (linear logic) "box". Asperti showed how the same problems might be approached from a categorical point of view [Asp95b], and Asperti and Laneve generalized the theory to the "interaction systems" [AL93]. The relations with the geometry of interaction are investigated in [ADLR94].

Sharing graphs present a problem of coherence. Suppose that the proof-net (or lambda-term) N reduces by standard cut-elimination (beta-reduction) to N'. Then, by reducing the sharing graph corresponding to N we do *not* obtain the sharing graph corresponding (in the given translation) to N'. The recovering of the proof-net N' is instead obtained by the so-called *read-back* process, a semantically based procedure *external* to the reduction system, which essentially computes the equivalence quotient of all the sharing graphs representing the same proof-net (term). A first contribution towards the solution of this problem is the notion of safeness in [Asp95]. In presence of certain safety conditions (which may be computed along the computation) some additional reductions may be performed, allowing a further simplification of the net.

We adopt, instead, a different approach. The main contribution of this paper is a solution to the coherence problem (for restricted proof-nets, see below) obtained by changing the way the information is coded into sharing graphs. This is achieved via two technical tools: (i) a new reduction rule (*absorption*) allowing a simplification of the net in some critical cases; (ii) a clear separation of the logical and control information in the representation of a net. The logical information takes the form of levels on the formulas of the proof-net; control is expressed by unifying fans and brackets into one single node (*mux*). It is this separation to allow the formulation of the absorption reduction and to enforce coherence.

Our results, like those of most of the literature, hold for restricted proof-nets, where weakening is not allowed. It should be clear that any approach to cut-elimination based on a *local* graph exploration may work only on connected components. If the syntax allows, during reduction, the creation of distinct components out of a single connected graph, then any local approach is bound to fail. This is why we ban weakening from our logic (cf. also [GAL92b]). A different solution is to allow weakening, but also to change the logic; e.g., take intuitionistic logic coded inside linear logic; this is (typed) λ-calculus, treated in [Gue95].

The insight needed to introduce our new techniques came from the proof theory of modal logics. In the context of proof-nets, the already mentioned notion of box is necessary to ensure soundness of the introduction of a modal connective (the of-course "!") and to allow the proper reduction of the proof-net during the cut-elimination process. A box is a global, explicitly given notion: each occurrence of an of-course connective in the proof-net "comes together" with a certain subgraph, its box. In [MM95]—applying to linear logic ideas and techniques previously developed for modal logic, see [MM96]—we discovered that a different, straightforward approach was possible, labelling with natural number indexes the formulas of the proof-net. The approach of [MM95], moreover, allowed a clear recognition, at any time, of the boundary of the box. This suggested our new, simple absorption rule. The approach has been applied to the optimal reduction of lambda terms in [Gue95], where the main algebraic techniques necessary to prove its correctness are developed. A generalization of the technique and detailed proofs may be found in Guerrini's thesis [Gue96].

Finally, we would like to attract the attention of the reader to our formalization of proof-nets as hypergraphs. We believe it an improvement over the current formulation, allowing a good mastering of the concepts and techniques needed in the field.

2 Levelled nets, proof-nets, reduction

We introduce in this section the net concepts we will use in the sequel. The most standard notions are that of restricted proof-structure and proof-net (Definitions 3 and 5; restricted in that weakening is not allowed), though given here as hypergraph (consistently with the presentation of [Gir87], but unlike most literature) and with levels instead of boxes (see [MM95] for the equivalence with the box approach). Proof-structures are special cases of *levelled-nets* (or ℓ-nets, Definition 1), which may contain additional links, responsible for duplication (*mux*'s and their duals *demux*'s; a single mux would correspond, in Lamping's approach, to one fan and several brackets). By *formula*, we mean a multiplicative-exponential linear logic formula; an *indexed* formula is a formula decorated with a non negative integer, the *level* of the formula.

Definition 1. An *ℓ-net* is a finite connected hypergraph whose nodes are labelled with indexed formulas and hyperedges (also called *links*) are labelled from the set $\{\text{cut}, \text{ax}, \wp, \otimes, !, ?\} \cup \{mux[i] |\ i \geq 0\} \cup \{demux[i] |\ i \geq 0\}$; the integer i in (de)muxes is the *threshold* of the link. Allowed links and nodes are drawn in Figure 1. The source nodes of a link are its *premises*; the target nodes are the *conclusions*. Premises and conclusions are assumed to be distinguishable (i.e., we will have *left/right* premises, i-th conclusion and so on), with the exception of ?-links. In an ℓ-net, each node must be conclusion of exactly one link and premise of at most one link; those nodes that are not premises of any link are the *net conclusions*; unary (de)muxes are also called *lifts*.

We assume that ℓ-net axioms have only atomic conclusions. Such a restriction does not decrease the expressive power of ℓ-nets. However, it would be possible to have a more economic representation of nets, allowing axioms with non exponential conclusions [Gue96].

Remark 2. Figure 2 states the correspondence between our ℓ-nets and the nets of [GAL92b, Asp95b] (see also Remark 7). A (de)mux with n auxiliary ports corresponds, in Asperti's notation, to a tree of fans with n leaves followed by

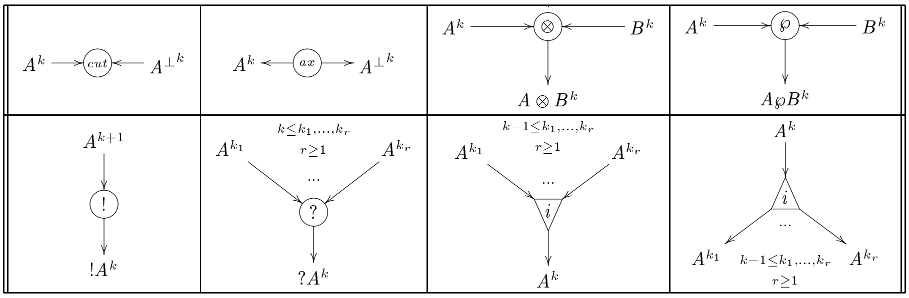

Fig. 1. ℓ-net links

a chain of brackets closed at the top by a croissant, one chain for each leaf. The length of the chain is given by the offsets of the corresponding port (i.e., the difference between the level of the formula assigned to such a port and the one assigned to the principal one) increased by 1. The top of Figure 2 shows the binary case (the triangle on the right side of the equivalence is then a fan and not a mux). A ?-link with a conclusion at level k corresponds to a bracket with an Asperti index equal to k (Gonthier index would be 0) followed by a configuration analogous to that of a mux with threshold k and conclusion at level $k+1$ (cf. the $\rhd_{exp}$ rule). The corresponding binary case is drawn at the bottom-left of Figure 2. An !-link is just a bracket indexed as the bottom bracket of a corresponding ?-link.

Definition 3. A *restricted proof-structure* is an ℓ-net without (de)muxes.

Definition 4. Let Π be a restricted proof-structure and let A^k be a premise of an !-link; we call *box* of A^k a sub-hypergraph $bx_\Pi[A^k]$ of Π verifying the following properties:

1. $A^k \in bx_\Pi[A^k]$ (A^k is the *principal door* of $bx_\Pi[A^k]$);
2. $bx_\Pi[A^k]$ is a restricted proof-structure;
3. each net conclusion of $bx_\Pi[A^k]$ different from the principal door is a premise, in Π, of a ?-link with conclusion at level $j < k$ (such ?-premises are the *secondary doors* of the box);
4. for each $B^j \in \Pi$, if $B^j \in bx_\Pi[A^k]$, then $j \geq k$.

We denote by $BX[\Pi]$ the set of boxes of Π. Because of the definition of ℓ-net, boxes are connected. By this we see that each !-premise node has at most one associated box.

Definition 5. Let Π be a restricted proof-structure. Π is a *restricted proof-net* iff all the net conclusions have level 0 and, after erasing all levels from the

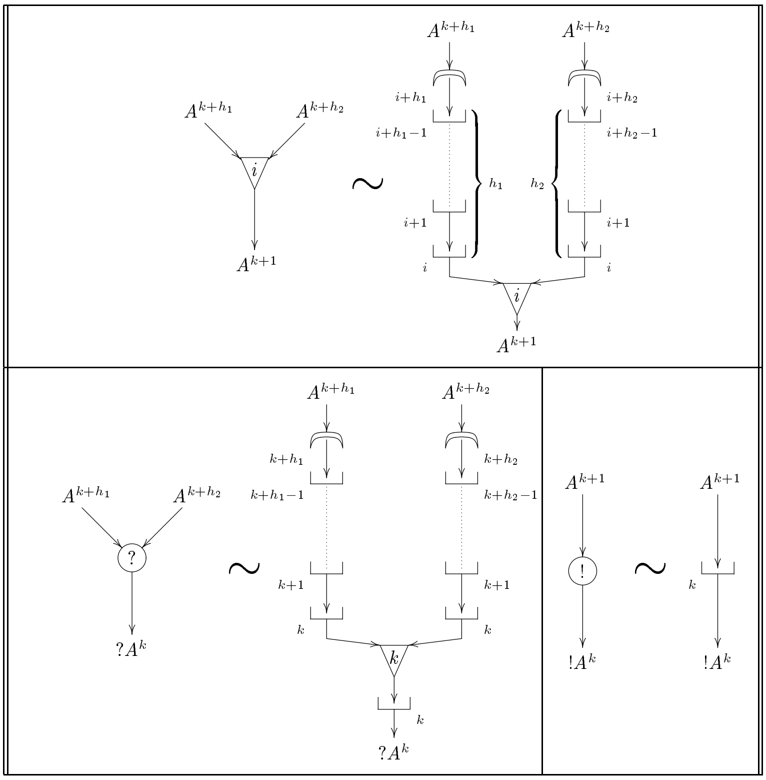

Fig. 2. Correspondence between ℓ-nets and sharing graphs

formulas, the resulting proof-structure *à la Girard*, equipped with $BX[\Pi]$ as set of boxes, verifies (one of) the correctness criteria [Gir87] for multiplicative-exponential proof-nets.

Theorem 6. *Let S be a weakening-free proof-net à la Girard, equipped with a given set of boxes, with conclusions $A_1, \ldots, A_m$. Then there is a restricted proof-net (as defined in this paper) S^* with conclusions $A_1^0, \ldots, A_m^0$, obtained by assigning to each node of S a level corresponding to the number of boxes that contain that node.*

2.1 Reduction

ℓ-nets may be used to implement a local and asynchronous version of the standard cut-elimination procedure (i.e., of the cut-elimination as defined in [Gir87]) for proof-nets. The elimination of *propositional* cuts (i.e., those formed by pairs tensor/par and axiom/cut) is directly mirrored in the corresponding rules. Figure 3 shows how to perform *standard exponential cut-elimination.* Observe, first, that the box $\Pi 1$ is (globally) duplicated. Second, after the reduction the different copies of $\Pi 1$ may have been put inside other boxes (this happens when the ?-node is a secondary door of another box). The notation $\Pi 1[k_i - k]$ means that all the levels of $\Pi 1$ have been incremented by $k_i - k$.

Levels and (de)muxes are designed to take care in a local way of both these aspects of the exponential reduction: multiple premises handle (incremental) duplication, while the threshold handles the (incremental) *reindexing* of the box—the re-computation of the new level of its nodes.

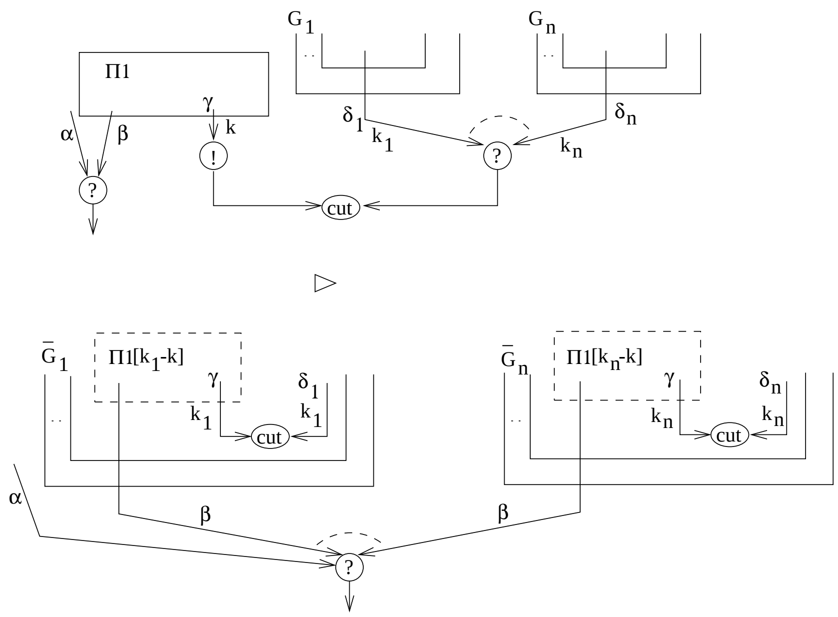

Fig. 3. Exponential cut reduction

We distinguish the rules in two kinds: the logical (or β) rules (Figure 4), where interaction happens through a cut-link (corresponding to a logical cut-elimination step); and the π rules (Figures 5, 6, and 7), when one of the interacting nodes is a mux/demux (corresponding to a step of incremental duplication and/or reindexing).

The set $\pi_{opt} = \pi - \triangleright_{dup}$ contains the only rules allowed during an *optimal* reduction (see Section 2.2). We stress the presence of the *absorption* rule ($\triangleright_{abs}$), corresponding to the case when the mux reaches the border of a box (through one of its secondary doors) and has therefore exhausted its job. It is motivated by the proof theoretical work in [MM95, MM96] (see also Section 5) and it is a special case of a safe reduction [Asp95].

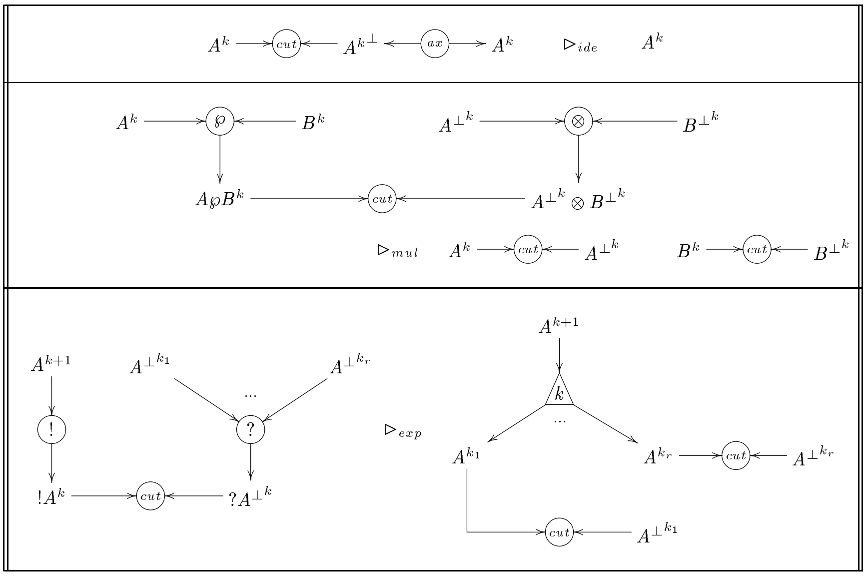

Fig. 4. Logical (or β) rules. The symmetric cases are not shown.

Remark 7. Any rule of π_{opt}, but $\triangleright_{abs}$, is admissible with respect to the reductions of [GAL92b] and the translation of Remark 2. The fact that $\triangleright_{abs}$ is not valid in that context depends on their choice to unify logical and control information in the same nodes, since in this way it is impossible to recognise in a local way whether a bracket configuration corresponds to a secondary door of a box (see also Section 5). If one sticks to the notation of [GAL92b], the solution is that indicated in [Asp95]: add another tag to each node, to record its "safeness".

Remark 8. Interactions between muxes are allowed only between mux/demux pairs in which the mux conclusion is the demux premise, i.e., connected through their principal ports as interaction nets (see π_{swap} and π_{anh}). Correspondingly, a non-identity logical link interacts with a demux when its conclusion (i.e., its principal port) is the demux premise (compare π_{odup} with π_{swap}). But, generalizing the rules present in [Asp95b], even a mux may interact with a logical link (see π_{dup}) when its conclusion is a premise of a logical link. An identity link behaves as a straight connection whose only purpose it to invert link orientation, so a cut-link interacts with a mux when one of its premise is a conclusion of the mux, and vice versa for the ax-link/demux case (see π_{idup}). The inversion implied by an identity link reflects in the mux/demux switching implied by a π_{idup} interaction.

Remark 9. It is impossible for a mux to reach a net conclusion. Let i be the threshold of a (de)mux, and let A^k be its (premise)conclusion; the fact follows

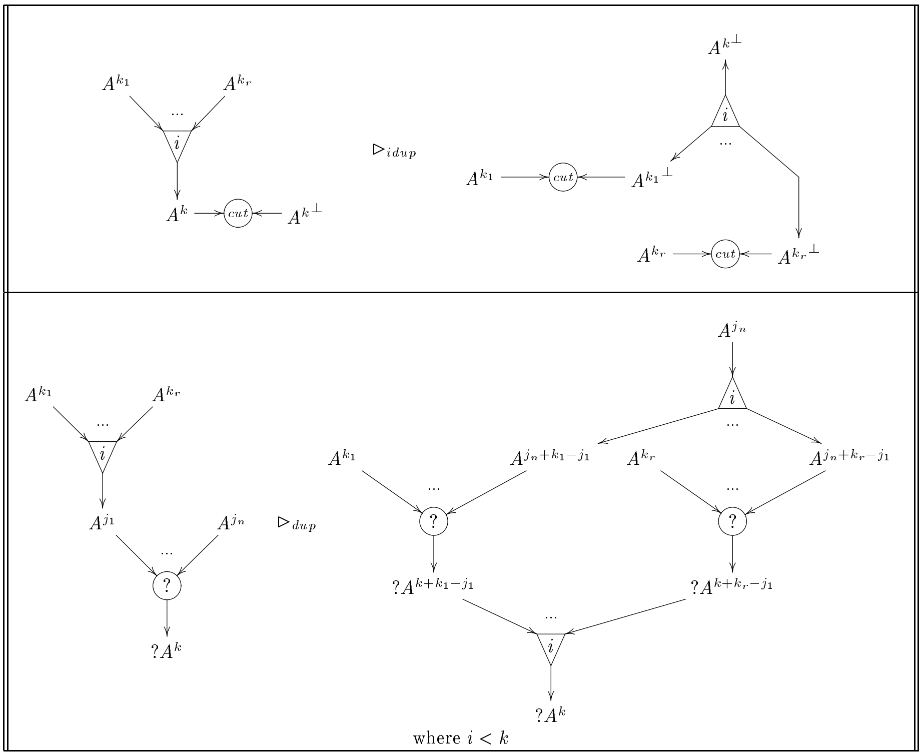

Fig. 5. Duplication rules, I. The following cases are not shown: in $\triangleright_{idup}$ the dual rule (demux/ax); in $\triangleright_{dup}$ the cases where interaction happens through another premise of the ?-link; in $\triangleright_{dup}$ the cases where the logical link is $\otimes$, $\wp$, !, instead of why-not (?).

since the relation $k > i$ is an invariant under reduction and any net conclusion has level 0.

2.2 Optimality

Optimality for the β reduction of λ-calculus has been studied by Lévy [Lév78, Lév80]. An analogous analysis may be performed in the case of proof-nets (see [GAL92b], or [AL93]). By a suitable labelling of (standard) proof-nets, we may define a Lévy labelled rewriting system for proof-nets, in which, as in the λ-calculus, the residuals of a cut have all the same label and new labels appear only when new cuts are created during reduction. Starting from a labelled proof-net N in which all nodes have different labels, we say that two cuts (not necessarily belonging to the same reduct of N) are in the same *Lévy family* iff they have the same label. A *family reduction* is a sequence of parallel rewritings $R_1 R_2 \ldots$

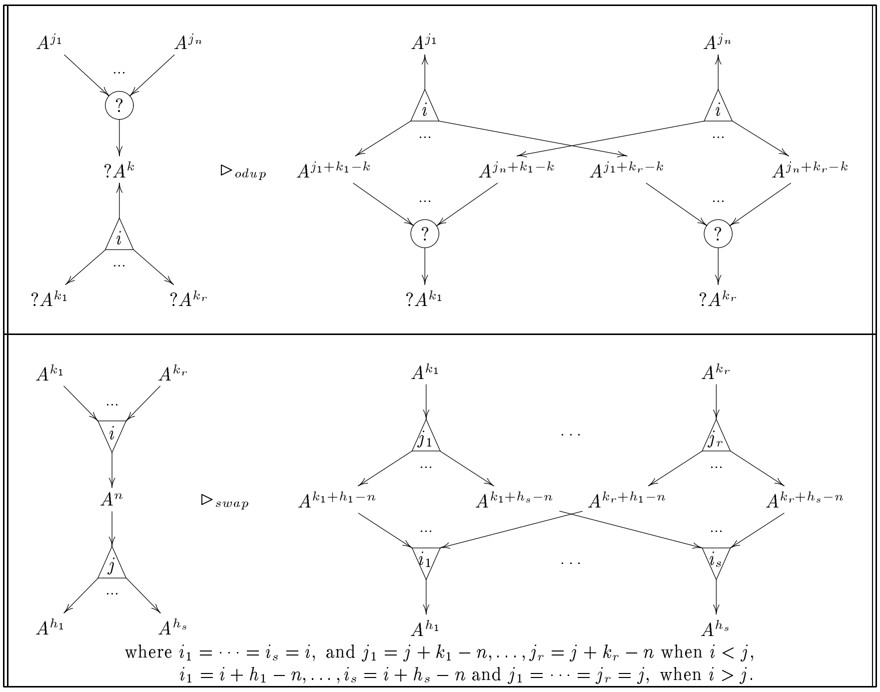

Fig. 6. Duplication rules, II. The following cases are not shown: in $\rhd_{odup}$ the cases where the logical link is $\otimes$, $\wp$, !, instead of why-not (?).

s.t. all the cuts in R_i are in the same family. A *complete* reduction is a sequence of rewritings where at each step are reduced *all* the cuts of the same family (i.e., if r and r' are two cuts in the same family, then $r \in R_i$ implies $r' \in R_i$). Finally, a *call-by-need reduction* of N is a sequence of rewritings where at any step we reduce at least a needed cut—a cut which appears (more precisely, a residual of which appears) in any rewriting sequence starting from N. Main argument of Lévy [Lév80] is that, in the case of λ-calculus, the optimal cost of the reduction of a λ-term may be taken as the number of β reductions of a call-by-need complete family reduction (in the λ-calculus case, a call-by-need strategy is left-most-outer-most). We assume the same measure (β contractions) for proof-nets.

It is important to observe that a redex of a restricted proof-net is always *needed*. This is not surprising, since without weakenings it is impossible to have redexes belonging to subgraphs that will be erased (i.e., not needed redexes). Therefore, all reduction strategies are effective call-by-need strategies.

We stress that the solution presented in this paper to the coherence problem is motivated by pure proof theoretical considerations. We have not studied the

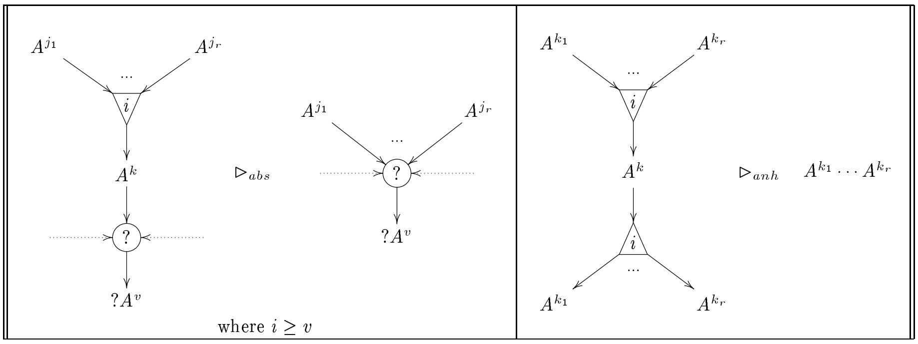

Fig. 7. Simplification rules

efficiency of our approach compared with the others approaches. Finding a good measure for the computational complexity of asynchronous and local reductions in proof-nets (and λ-calculus) is an important open problem, outside the scope of the present paper (e.g., [Asp96, LM96]).

3 Coherence

We state in this section our main results, namely that the reduction rules $\beta + \pi$ solve the coherence problem for ℓ-nets. This is not trivial, since the rules may be fired in any order (logical and non-logical reductions will be in general interleaved). Our proof strategy, analogous to the one used in [Gue95] for the λ-calculus, is to simulate ℓ-nets over proof-nets. Via a (long and technical) detour through an algebraic semantics for ℓ-nets we prove the following main theorems. For the sake of their statement, let an ℓ-net G be *correct* iff there exists a restricted proof-net N s.t. $N \rhd^* G$; informally, an ℓ-net is correct if it represents a restricted proof-net.

Theorem 10. *Let G be a correct ℓ-net.*

1. *The π rules are strongly normalizing and confluent on G. The π normal form of G is a restricted proof-net.*
2. *The $\beta + \pi$ rewriting rules are strongly normalizing and confluent on G. The $\beta + \pi$ normal form of G is a restricted proof-net.*
3. *The π normal form of G reduces by standard cut-elimination to its $\beta + \pi$ normal form.*

Theorem 11. *The $\beta + \pi_{opt}$ rewriting rules are Lévy optimal.*

Confluence of $\beta + \pi$ implies thus the following.

Theorem 12. *Let G be a correct ℓ-net and N be its $\beta+\pi$ normal form. Let G' be a $\beta+\pi_{opt}$ normal form of G, then $G' \rhd^*_\pi N$.*

Definition 13. The *read-back* $\mathcal{R}(G)$ of a correct ℓ-net G is the π normal form of G.

By Theorem 12, normalization of correct ℓ-nets may be performed in two distinct steps: first optimal reduction ($\beta+\pi_{opt}$), then read-back reduction (π).

4 The inside of ℓ-nets

We give in this section a more detailed overview of the way the results of the previous section are obtained.

The technical core of the approach is an algebraic semantics of ℓ-nets, which cannot be presented here for lack of space. It may be found in [Gue96], which we refer to also for the statements below without a proof.

4.1 Sharing morphisms

Definition 14. An *s-morphism* is a surjective graph homomorphism $M : G_0 \to G_1$ of ℓ-nets which preserves the link and node labels (i.e., the type of links and the levels of nodes) and the names of the link ports, and is injective when restricted to the net-conclusions.

If $M : G_0 \to G_1$, then G_1 is equal in all respects to G_0 but for the number of premises of (de)mux and ?-links (e.g., a k-ary mux may be mapped to one with $k' \geq k$ premises). G_0 may thus be seen as a "less shared instance" of G_1; we write $G_0 \prec G_1$ when there exists an s-morphism from G_0 to G_1.

We need the concept of *uℓ-net*, an ℓ-net whose (de)muxes have a single (conclusion)premise. Over uℓ-nets we define a notion of reduction that lives midway between standard proof-net reduction (global duplication + boxes redrawing) and ℓ-net reduction. While the π rules are unchanged (though always applied with unary muxes), the β rule for the exponential cut is formulated with a global duplication of boxes (computed using the algebraic semantics to extend Definition 4 to uℓ-nets) without any modification of the involved levels, and the introduction of a lift for any duplicated box. We write $G \rhd_{\beta_u} G'$ for *the uℓ-net G reduces, by a β uℓ-net reduction, to the uℓ-net G'.*

We will use uℓ-net reduction to simulate ℓ-net reduction. The crucial properties of uℓ-nets are expressed in the following statements (see [Gue96] for proofs and more details).

Proposition 15. *If G is a correct ℓ-net, then there is a unique correct uℓ-net G^u (the* least shared instance *of G) such that $G^u \prec G$.*

Proposition 16. *Let G be a correct uℓ-net.*

1. *There is no infinite π reduction of G.*

2. G has a unique π normal form, which we indicate with $\mathcal{R}(G)$.
3. $\mathcal{R}(G)$ is a restricted proof-net.

Proposition 17. *Let G be a correct uℓ-net and let $G \rhd_{\beta_u} G_1$. The uℓ-net G_1 is correct and $\mathcal{R}(G) \rhd \mathcal{R}(G_1)$ by the standard proof-net reduction of the same redex.*

Strong normalization of standard proof-net reduction of (restricted) proof-nets implies thus the following.

Corollary 18. *The reduction relation $\pi + \beta_u$ is strongly normalizing on correct uℓ-nets.*

4.2 Correctness of ℓ-net reduction

We may now simulate π and β reduction of ℓ-nets over uℓ-nets.

Lemma 19. *Let G_0 be a correct ℓ-net for which there exists a correct uℓ-net G_0^u s.t. $G_0^u \prec G_0$. If $G_0 \rhd_\pi G_1$, there exists $G_0^u \rhd_\pi^+ G_1^u$ s.t $G_1^u \prec G_1$.*

Proof. (Sketch) Let M be the s-morphism between G_0^u and G_0 and let r be a redex of G_0. The counterimage $M^{-1}(r)$ of r w.r.t. M is a set of redexes that may contain only a case of critical pair: two lifts pointing to the premises of the same ?-link. If the redex r is a duplication, the algebraic semantics (note that G_0^u is correct) allows to prove that such two lifts must be equal and then that such a critical pair is confluent. Hence, let us execute in any order the redexes of G_0^u in $M^{-1}(r)$ (closing as previoulsy stated the critical pairs present in it), the result is G_1^u. It is also not difficult to see that the s-morphism between G_1^u and G_1 maps any residual of a link v of G_0^u to the residual of $M(v)$.

Proposition 20. *Let G be a correct ℓ-net with least shared instance G^u.*

1. There is no infinite π reduction of G.
2. G has a unique π normal form. Moreover, this π normal form is $\mathcal{R}(G^u)$.

Proof. (Sketch) (1) By Proposition 16, G^u strongly normalizes by π reduction to the restricted proof-net $\mathcal{R}(G^u)$. By Lemma 19, the existence of an infinite π reduction of G would contradict the strong normalization of G^u.
(2) The π normal form $\mathcal{R}(G^u)$ of G^u does not contain lifts (by Proposition 16 it is a restricted proof-net). Let now N be any π normal form of the ℓ-net G. Observe that $\mathcal{R}(G^u) \prec N$, by Lemma 19, and thus also N does not contain muxes, for a sharing-morphism preserves links. Hence it is an uℓ-net. Some thought shows that this implies $\mathcal{R}(G^u) = N$.

From the previous lemma we get Theorem 10(1). We now simulate ℓ-net β reduction on β_u reduction of uℓ-nets.

Proposition 21. *Let G_0 be a correct ℓ-net for which there exists a correct uℓ-net G_0^u s.t. $G_0^u \prec G_0$. If $G_0 \rhd_\beta G_1$, there exists $G_0^u \rhd_{\beta_u}^+ G_1^u$ s.t G_1^u is correct and $G_1^u \prec G_1$.*

Proof. (Sketch) Let $M : G_0^u \to G_0$ and let r be a β redex of G_0. The unshared reduction corresponding to the reduction of r is a development of the set of redexes $M^{-1}(r)$ (a development of a set of β redex of a proof-net is the analougus of a development of a set of β redexes for the λ-calculus). The s-morphism between the uℓ-net G_1^u obtained in this way and G_1 is the one mapping any residual of a link l of G_0^u to the residual of its image $M(l)$ (see the detailed proof given in [Gue96] for the λ-calculus case).

We finally prove Theorem 10(2-3).

Theorem 22. *Let G be a correct ℓ-net.*

2. *The $\beta + \pi$ rewriting rules are strongly normalizing and confluent on G. The $\beta + \pi$ normal form of G is a restricted proof-net.*
3. *The π normal form of G reduces by standard cut-elimination to its $\beta + \pi$ normal form.*

Proof. (Sketch) Assume $G \rhd_\beta^* G_1 \rhd_\pi^* G_2$, and let G^u be the least shared instance of G. By 21 and 19 we have a corresponding unshared reduction $G^u \rhd_{\beta_u}^* G_1^u \rhd_\pi^* G_2^u$. Moreover, by 17 and 21, $\mathcal{R}(G) \rhd^* \mathcal{R}(G_1)$ by a standard proof-net reduction. By Proposition 20, $\mathcal{R}(G_1) = \mathcal{R}(G_2)$, and the π reduction of the previous pair cannot be infinite. Thus, any reduction of G may be decomposed in an alternating sequence of a non empty β reduction and of a finite number of π rewritings. Since each element of such a sequence corresponds to a non-empty sequence of β standard proof-net rewritings, the alternating sequence cannot be infinite, for otherwise we would have an infinite reduction of a proof-net. This establishes both strong normalization and (3). Moreover, we conclude the uniqueness of the $\beta + \pi$ normal form and then the confluence of $\beta + \pi$. In fact, by Proposition 16 and Proposition 20 we see that any $\beta + \pi$ normal form of G is a restricted proof-net, that is, it is the (unique) normal form of $\mathcal{R}(G)$ under standard proof-net reduction.

5 Conclusions

We have presented in this paper a solution to the coherence problem for the sharing graph representation of (restricted) proof-nets and their computations. This result has been made possible by a change in the representation of the nets. As discussed in Remarks 2 and 7, there is a rather simple correspondence between our approach (levels on formulas and only one kind of control nodes—(de)muxes) and the one established in the literature (levels on nodes, two kinds of control nodes—fans and brackets). This shift of notation, however, is crucial and responds to a deep conceptual issue: separating logic from control. The level of a formula, indeed, is a logical information, necessary to ensure not only the correctness of the reduction, but even the static correctness of a net. This has been clear since our previous work on levelled approaches to modal and linear proof theory [MM95, MM96]. In the case of this paper, levels belong to the logic and (de)muxes and their reduction rules belong to control. It is this separation to make coherence possible. In the standard approach, instead, logic and control are blurred together. There is more uniformity of notation, but the price to be paid is the difficulty to recognize in a local way the border of boxes, that is to eventually guarantee coherence. A different solution is that of the safe reductions of [Asp95], of which our absorption is a special case.

It remains to address the problem of full proof-nets, where weakening is allowed. Weakening in linear logic can produce boxes whose contents are disconnected. Such boxes can be also generated by the cut-elimination procedure, even starting from proof-nets whose boxes are connected. The crucial case is that of a box whose principal door has as premise a weakening link, and hence it needs a separate component S (that must be a proof-net) to be a valid conclusion of the box. This separate component yields the secondary doors of the box. Now, any attempt to reindex/duplicate the box through its principal door will not reach the disconnected net S. Observe that this problem is shared by all the approaches proposed so far, as any local graph rewriting procedure cannot deal with disconnected components. There is a simple way to bypass this problem, e.g., by restricting the proof-net syntax to generate interaction systems (this means for example to be able to code typed λ-calculus, intuitionistic linear logic and so on). We claim, however, that a solution to the general case will call for an extension of the proof-net syntax in order to avoid the formation of disconnected boxes. Research in this direction is ongoing.

Acknowledgments

We are happy to thank Andrea Asperti, for the many discussions, Laurent Regnier, for many detailed remarks on the subject of this paper, and an anonymous referee, for help in improving presentation.

References

[ADLR94] Andrea Asperti, Vincent Danos, Cosimo Laneve, and Laurent Regnier. Paths in the lambda-calculus: three years of communications without understanding. In *Proc. of 9th Symposium on Logic in Computer Science*, pages 426–436, Paris, France, July 1994. IEEE.

[AL93] Andrea Asperti and Cosimo Laneve. Interaction systems. In *Int. Workshop on Higher Order Algebra, Logic and Term Rewriting*, 1993.

[Asp95b] Andrea Asperti. Linear logic, comonads and optimal reductions. *Fundamenta infomaticae*, 22:3–22, 1995.

[Asp95] Andrea Asperti. $\delta \circ !\varepsilon = 1$: Optimizing optimal λ-calculus implementations. In Jieh Hsiang, editor, *Rewriting Techniques and Applications, 6th International Conference, RTA-95*, LNCS 914, pages 102–116, Kaiserslautern, Germany, April 5–7, 1995. Springer-Verlag.

[Asp96] Andrea Asperti. On the complexity of beta-reduction. In *POPL*, St. Petersburg Beach, Florida, 1996. ACM.

[GAL92a] Georges Gonthier, Martín Abadi, and Jean-Jacques Lévy. The geometry of optimal lambda reduction. In *Proc. of Nineteenth Principles of Programming Languages (POPL)*, pages 15–26. ACM, January 1992.

[GAL92b] Georges Gonthier, Martín Abadi, and Jean-Jacques Lévy. Linear logic without boxes. In *Proc. of 7th Symposium on Logic in Computer Science*, pages 223–234, Santa Cruz, CA, June 1992. IEEE.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Gue95] Stefano Guerrini. Sharing-graphs, sharing-morphisms, and (optimal) λ-graph reductions. In Z. Khasidashvili, editor, *1st Tbilisi Symposium on Logic, Language, and Computation*, CSLI Lecture Notes (to appear). Tbilisi, Georgia, October 1995.

[Gue96] Stefano Guerrini. *Theoretical and Practical Aspects of Optimal Implementations of Functional Languages.* PhD thesis, Dottorato di Ricerca in Informatica, Pisa–Udine, January 1996.

[Lam90] John Lamping. An algorithm for optimal lambda calculus reduction. In *Principles of Programming Languages (POPL)*, pages 16–30. ACM, 1990.

[Lév78] Jean-Jacques Lévy. *Réductions Correctes et Optimales dans le lambda-calcul.* PhD Thesis, Université Paris VII, 1978.

[Lév80] Jean-Jacques Lévy. Optimal reductions in the lambda-calculus. In Jonathan P. Seldin and J. Roger Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 159–191. Academic Press, 1980.

[LM96] Julia L. Lawall and Harry G. Mairson. Optimality and inefficiency: What isn't a cost model of the lambda calculus? In *1996 ACM International Conference on Functional Programming*, 1996.

[MM96] Simone Martini and Andrea Masini. A computational interpretation of modal proofs. In H. Wansing, editor, *Proof theory of Modal Logics.* Kluwer, 1996. To appear.

[MM95] Simone Martini and Andrea Masini. On the fine structure of the exponential rule. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, volume 222 of London Mathematical Society Lecture Note Series, pages 197–210. Cambridge University Press, 1995. Proceedings the 1993 Cornell Linear Logic Workshop.