Strong Normalization Property for Second Order Linear Logic

Michele Pagani[a,1], Lorenzo Tortora de Falco[b]

[a]*Dipartimento di Informatica – Università di Torino*
Corso Svizzera 185, 10149 Torino, Italy
[b]*Dipartimento di Filosofia – Università Roma Tre*
Via Ostiense 231, 00144 Roma, Italy

Abstract

The paper contains the first complete proof of strong normalization (SN) for full second order linear logic (LL): Girard's original proof uses a standardization theorem which is not proven. We introduce sliced pure structures (sps), a very general version of Girard's proof-nets, and we apply to sps Gandy's method to infer SN from weak normalization (WN). We prove a standardization theorem for sps: if WN without erasing steps holds for an sps, then it enjoys SN. A key step in our proof of standardization is a confluence theorem for sps obtained by using only a very weak form of correctness, namely acyclicity slice by slice. We conclude by showing how standardization for sps allows to prove SN of LL, using as usual Girard's reducibility candidates.

Key words: (weak strong) normalization, confluence, standardization, linear logic, proof-nets, additive connectives, sliced pure structures

1. Introduction

In every abstract approach to computation, the distinction between terminating and non-terminating processes is crucial. A rewriting system enjoys *weak normalization* (WN) if every term of the system can be executed in a finite number of steps.

In the λ-calculus, non terminating computations start from λ-terms that strongly exploit *self-application*: every λ-term can be applied to itself (see for example [13]). Termination fails for the λ-calculus (even in its weak form WN), but holds for some of its most remarkable subsystems: the simply typed λ-calculus and its extension Girard's system F ([6]). The proofs of WN for these calculi have a deep logical content: they correspond to proofs of consistency in the logical sense, as highlighted by the *proofs-as-programs* paradigm. This paradigm is also called *Curry-Howard isomorphism* and establishes a correspondence between a fragment of intuitionistic natural deduction

Email addresses: pagani@di.unito.it (Michele Pagani), tortora@uniroma3.it (Lorenzo Tortora de Falco)

[1]This work has been supported by the postdoc fellowship "Ricerche sulla geometria della logica", Dipartimento di Filosofia, Università Roma Tre

and the typed λ-calculus, so that basically: (i) a type can be seen as a formula (and vice versa); (ii) a λ-term can be seen as a proof (and vice versa); and (iii) the β-reduction of a λ-term can be seen as the application of the cut-elimination procedure to the corresponding proof (and vice versa). The β-redexes correspond to the cuts of the natural deduction (i.e. to the pairs of an introduction rule and an elimination rule of the same connective [19]), hence the β-reduction to a normal form is equivalent to a strategy for eliminating the cuts in a proof: WN corresponds exactly to cut-elimination. In traditional proof-theory (dating back to Gentzen) cut-elimination is a key property of a logical system, from which the consistency of the system immediately follows[2]: WN of the simply typed λ-calculus (resp. of system F) corresponds then to a consistency proof for the implicational fragment of intuitionistic natural deduction NJ (resp. second order natural deduction NJ^2).

The proofs-as-programs approach is a way to put constraints on the possibility of building λ-terms, and more precisely on self-application: from the logical point of view, these constraints are sufficient conditions to prove the consistency of the corresponding logical system[3]. While in the simply typed λ-calculus self-application is simply forbidden, in system F a (weak) form of self-application is accepted. System F combines then computational strength and termination (here is the remarkable interest of WN for F). Gödel's second incompleteness theorem sheds then a new light on the "difficulty" (and the deep meaning) of the termination property. Indeed, Peano Arithmetic PA can be translated into NJ^2, so that WN for F cannot be proven within NJ^2 (and of course this holds for any subsystem of the λ-calculus whose corresponding logical system contains PA).

The terms of system F actually enjoy *strong normalization* (SN), a much stronger termination property than WN: whatever execution strategy one applies to F's terms one eventually reaches a normal form. SN corresponds to saying that the tree of the computations starting from a term is well-founded. Several techniques to infer SN from WN have been proposed (see for example [21]). In [8] (p.150-159), the author adapted Gandy's proof for Gödel's system T (see [5]) to NJ. The general method proposed by Gandy can be applied to any logical system S as follows:

1. modify S in such a way that "nothing is lost" during normalization[4] (rewriting steps never erase pieces of proofs): let's denote by $\xrightarrow{\neg \mathsf{e}}$ the rewriting rule of S "without erasing steps";
2. prove WN for $\xrightarrow{\neg \mathsf{e}}$ (we 'll often write in the sequel that S enjoys $\mathrm{WN}^{\neg \mathsf{e}}$) and confluence[5] for $\xrightarrow{\neg \mathsf{e}}$;

[2]Things went actually the other way round: cut-elimination was proven by Gentzen *in order to* prove consistency of Peano Arithmetic PA.

[3]A term associated with a proof in a logical system is often said to be *typed*, the conclusion of the proof is a *type* of the term.

[4]Thanks to the Curry-Howard isomorphism one can use the language of λ-terms for proofs, and vice versa.

[5]A rewriting system enjoys the confluence property when if $t \xrightarrow{*} t_1$ and $t \xrightarrow{*} t_2$, there always exists t' such that $t_1 \xrightarrow{*} t'$ and $t_2 \xrightarrow{*} t'$, where t, t_1, t_2, t' are terms and $\xrightarrow{*}$ is the reflexive and transitive closure of the rewriting rule of the system, see Section 2.2.

3. define on the proofs of S a size function $|\cdot|$, which is strictly increasing with respect to $\xrightarrow{\neg e}$;
4. conclude that S enjoys SN.

Notice that the size function $|\cdot|$ "computes" an upper bound for the length of the $\xrightarrow{\neg e}$ reduction sequences starting from any proof π of S: it is the value of $|\cdot|$ on the $\neg e$-normal form of π.

Linear Logic (LL [7]) is a refinement of intuitionistic logic and of classical logic, which gives a *logical* status to the operations of erasing and copying (corresponding to the *structural rules* of intuitionistic logic and of classical logic). This change of viewpoint on structural operations has striking consequences: one of the most important is the introduction of proof-nets (simply called *nets* in this paper). Proof-nets yield a graph-theoretical representation of computation, where the strict distinction between inputs and outputs completely vanishes (this is a sharp difference between nets and λ-terms). In LL, cut-elimination is defined directly for proof-structures (general graphs which are not necessarily proofs), and nets are the "correct" proof-structures (the ones satisfying a *correctness* condition). The presence of proof-structures as new computational objects widens the space of possible interactions between logical agents: this makes the system much richer and more complicated. We'll show how, for the termination property of LL proof-structures, not only the distinction typed/untyped is relevant (this was already the case for λ-terms), but also the one correct/noncorrect (which is independent from the typed/untyped one). Indeed this idea underlies much of Girard's work, since [7]: the correspondence between logical correctness and termination property.

We present the first complete proof of SN for full second order LL: we prove SN for Girard's nets as presented in [24]. Because SN of full second order LL entails the consistency of (second order) PA, its proof cannot be carried out within PA: it uses Girard's reducibility candidates (introduced in [6] and already used for LL in [7]). However, by applying Gandy's method to LL, we show that strong principles are needed only to prove WN (more precisely $\mathrm{WN}^{\neg e}$).

Some readers might be surprised that so many years after the discovery of LL (and after so many papers on the subject) no complete proof of SN for LL has been given. In [7] Girard gives a "proof" of SN based on a theorem called *standardization theorem* (theorem 4.25 p.72 of [7]), which is not proven in the paper[6]. Intuitively, the theorem states that if a net can be transformed into a $\neg e$-normal form "without applying erasing steps", then it enjoys SN. This basically corresponds to achieving all the tasks of Gandy's method, except the proof of $\mathrm{WN}^{\neg e}$.

In LL, the absence of a unique output (actually the vanishing of the distinction between inputs and outputs) entails the absence of a distinguished cut (something like a "head cut"). There still exists some kind of hierarchy on cuts in LL nets (namely the

[6]Let us point out here that the word "standardization" used by Girard might be misleading w.r.t. the λ-calculus literature: the "standardization property" of the λ-calculus refers to a different well-known theorem, while Girard's standardization theorem is called *conservation theorem* (see e.g. [1]); moreover, Girard's standardization is called "propriété de striction" by Danos in [2] (théorème 8.31 p.64). In the present paper we will adopt Girard's terminology, however.

so-called *exponential depth*), but no difference can be made between two cuts with the same depth. Worse, the reduction of a cut may seriously affect the status (and thus the potential reduction) of other cuts at the same exponential depth. There is no analogue of such a meddling of cuts in the life of their fellows with the same exponential depth in the λ-calculus: the β-reduction of a redex may affect other redexes only if these last ones are deeper than the former one (in LL terminology: the affected redexes must have a greater exponential depth than that of the reduced one). The reader can refer to Subsection 3.2 (where the notion of *exponential dependence* is introduced to overcome this difficulty) for a more precise discussion. All this makes the standardization theorem an essential ingredient of the SN proof for second order LL, which *is not* a straightforward adaptation of the WN proof. This is in sharp contrast with what happens for Girard's system F (see [6]). The presence of an head redex in F's terms makes it easy to turn the proof of WN into a proof of SN (by a slight modification of the definition of reducibility candidate): the proof of SN is an easy variant of the proof of WN in system F. One might find more elegant (following Gandy) to distinguish the logical part of the SN proof (the proof of WN) from the purely combinatorial part (the proof of the standardization theorem) but in F it is still possible to mix them without loosing control on the combinatorial part of the proof. This becomes very difficult in LL, where a subtle theorem of standardization is needed to turn WN into SN. Something similar to what we have in LL can be found in λ-calculi with explicit substitution (see for example [4]).

In [2], Danos proves standardization (théorème 8.31 p.64) for second order multiplicative and exponential LL (MELL^2), a significant fragment of LL containing system F: Danos' result achieved the proof of SN for MELL^2. Later on, SN of several other classical and linear systems has been proven thanks to appropriate embeddings in MELL^2 (see for example [3] and [15]). But up to now, no proof of SN was available for the full system, essentially because of the presence of the additive connectives, whose computational behaviour is difficult to handle (for example, cut-elimination is not confluent in presence of the additives, at least in the traditional syntax). The main goal of the paper is to finally fill this (rather big) gap in LL's literature. For the sake of completeness, we mention here two previous attempts to prove this result: annexe A of [23] and [18]. In annexe A of [23] a proof of a variant of the standardization theorem is given but, as explained in that paper, it is not sufficient to prove SN for full LL. In [18], a nice approach to termination using phase semantics is proposed, which is suitable for WN (at least in the fragment MELL^2) but not for SN: on the one hand the proof of standardization for MELL^2 is not convincing (it is only "sketched" as the author writes) and on the other hand in presence of the additives the considered cut-elimination procedure is not the full one (see Subsection 5.2 for a detailed discussion on [18]).

Our approach is to start from scratch, having in mind the two following guidelines:

- as soon as the system is powerful enough, the combinatorial part of the SN proof (standardization or any of its variants) should be split from the part involving logically strong principles (WN or any of its variants);
- it should be stressed where logic (more precisely types and correctness) comes

into the picture: to which extent is it possible to compute with untyped proof-structures?

Our main contribution is actually a standardization theorem (Theorem 4.2), proven for "sliced pure structures" (sps) — a notion of proof-structure which yields a better account of additive cut-elimination than the nets of [7]. Section 5 studies the translation from nets to sps, proving in particular that a net is SN whenever its translation into sps is SN. The notion of "slice" was introduced in [7] in order to reduce the difficulty of dealing with the additive connectives. In the polarized framework, its variant "sliced proof-structure" is proven to enjoy nice properties [16]. The main reason why we use slices in this paper is that we need to prove a confluence property (remember it is part of Gandy's method), and it is the only known syntax of LL for which one can hope to prove such a property. Our sps are an extension of sliced proof-structures to full *untyped* LL. Indeed, following the idea that types and correctness should be used only when necessary, we start our analysis with the most general kind of graph we have: untyped and noncorrect. We show in Section 2 how one can always compute with such general structures. This is a first novelty of our paper: both in [7] and [24], in presence of the additives, computations are defined only for nets (typed and correct proof-structures). However, without any kind of correctness, computation behaves very badly (and this is not related to the presence/absence of types): we give a counterexample to both WN and confluence for such general (noncorrect) sps (see Figure 12). We then introduce the correctness condition "acyclicity slice by slice" (AC condition, Definition 2.15). It is well-known that this condition is far too weak to characterize proofs (see [10]), and there is a wealth of papers presenting criteria stronger than the correctness slice by slice in order to characterize logical correctness (see for instance [10, 11]). What we show here is that, despite its weakness, the AC condition is sufficient to prove confluence (Theorem 4.18) and standardization (Theorem 4.2): from the computational point of view only a very weak form of correctness is needed.

Let us now describe more precisely the structure of the paper. As the connection between the different results is rather delicate and complex, we conclude the introduction by a graphical representation of the logical structure of the paper (see Figure 1). Also, the whole paper is scattered with examples and figures: we hope this will help the reader.

Section 2 is devoted to sps. In Subsection 2.1 we define sps (Definition 2.2) and their cut-elimination (Definition 2.12). Subsection 2.4 is entirely devoted to motivating our results and choices by several examples and counterexamples. In Subsection 2.5 we introduce the weak form of correctness used in the paper and expressed by the AC condition of Definition 2.15.

Section 3 achieves a first essential step of our proof: Definition 3.1 splits the cut-elimination rewriting rule (denoted by $\xrightarrow{\texttt{cut}}$) into two strongly normalizing reductions, the logical reduction ($\xrightarrow{\texttt{log}}$, proven to enjoy SN in Proposition 3.4) and the structural reduction ($\xrightarrow{\texttt{str}}$, proven to enjoy SN in Proposition 3.10). These two reductions are disjoint and their union is $\xrightarrow{\texttt{cut}}$; they both enjoy SN on sps satisfying AC, even if their union $\xrightarrow{\texttt{cut}}$ does not even enjoy WN (the untyped λ-calculus can be embedded in sps satisfying AC, see in particular the example of Fig. 10). Propositions 3.4 and 3.10 are

essential ingredients in the proof of confluence (Theorem 4.18) of the next section.

In Section 4 we apply Gandy's method to sps: we distinguish the erasing cuts from the non erasing ones and we define the $\xrightarrow{\neg e}$ rewriting rule for sps (Definition 4.1). We then explain (Subsection 4.1) why SN of $\xrightarrow{\neg e}$ entails SN of $\xrightarrow{cut}$: a postponement lemma (Lemma 4.4) allows to "delay" erasing steps after any non-erasing one, from which the result easily follows (Proposition 4.5). Subsection 4.2 shows that from WN of $\xrightarrow{\neg e}$ one can deduce SN of $\xrightarrow{\neg e}$: the key point here is the confluence theorem (Theorem 4.18), proven for a *labelled* version of sps (see Definitions 4.6 and 4.7). We first prove that labels allow to define an increasing size on (labelled) sps[7], and that (assuming confluence holds) this allows to prove $\mathrm{WN}^{\neg e} = \mathrm{SN}^{\neg e}$ for sps satisfying AC: from this equality and Subsection 4.1, our main result (standardization, Theorem 4.2) immediately follows. The rest of Section 4 is devoted to the proof of confluence, based on the following results:

1. confluence of the labelled version of the logical reduction $\xrightarrow{log}$ of Section 3: it follows from local confluence (Lemma 4.12) and SN (immediate consequence of Proposition 3.4 of Section 3);
2. confluence of the labelled version of the structural reduction $\xrightarrow{str}$ of Section 3: it follows from local confluence (Lemma 4.14) and SN (immediate consequence of Proposition 3.10 of Section 3);
3. commutation of the labelled versions of $\xrightarrow{log}$ and $\xrightarrow{str}$ (Lemma 4.17);
4. Hindley-Rosen lemma (see [12]): a rewriting rule which is the union of two confluent rewriting rules which commute is itself confluent.

It is important to stress the fact that confluence (so as the main result of the paper, Theorem 4.2) is far from being an immediate consequence of the same result for MELL. It is only partially true that sps allow to work "slice by slice": the additive commutative normalization step (the nightmare of normalization in presence of the additives, see Section 5 and Figure 19) is not explicitely present in our syntax, but it is hidden in other normalization steps (the $(!/?d)$ and the $(!/!)$ steps of Definition 2.12). Indeed the exponential connectives are the bridge between the additive and the multiplicative worlds through the isomorphism $!(A\&B) = !A \otimes !B$.

Last, Section 5 introduces the nets of [24] with units, and it turns a proof of $\mathrm{WN}^{\neg e}$ for sps into a proof of SN for the nets of [24]. The syntax of [24] is described in Subsection 5.1 and it generalizes Girard's notion of net, [7]. As already mentioned, confluence (even local confluence) fails for nets, hence to apply Gandy's method one must pass through a confluent syntax, in our case the syntax of sps. Subsection 5.2 contains the motivations for our choice of sps and relates our results to previous attempts (namely [7] and [18]). Subsection 5.3 gives a translation of nets into sps, called *slicing*, and it shows that the slicing of a net β satisfies AC (Proposition 5.1), and that β enjoys SN whenever its slicing does (Proposition 5.6). Finally in Subsection 5.4 we prove that the slicing of any net enjoy $\mathrm{WN}^{\neg e}$ (Theorem 5.11). Standardization for sps (Theorem 4.2) allows then to conclude SN for nets (Theorem 5.12). Actually, the proof

[7] Notice that the AC condition is not needed here.

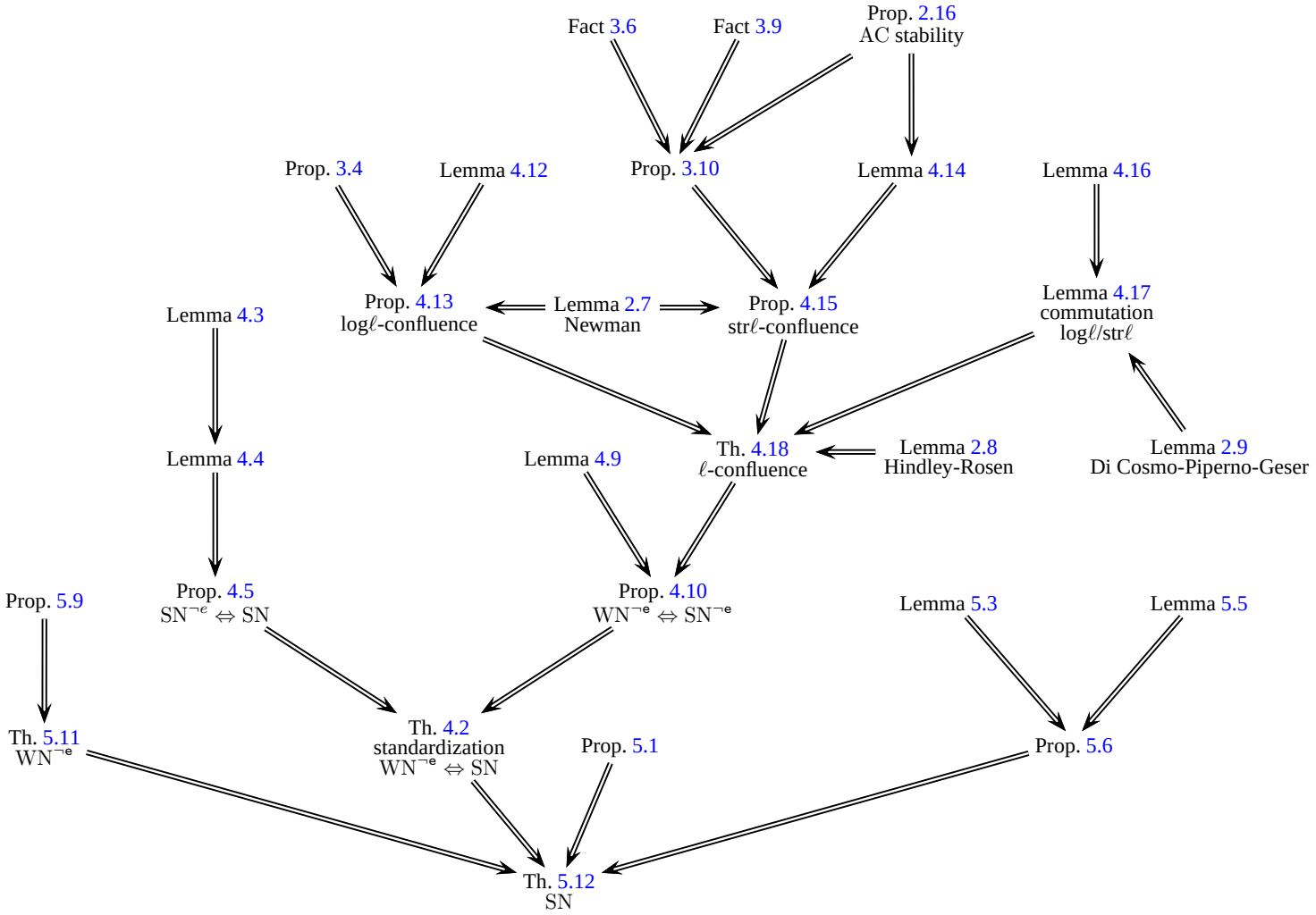
of Theorem 5.11 is given in a terse and sketchy style (using a simple variant of Girard's reducibility candidates): this is because there is no doubt about this result, and no real point in giving more details in the present paper.

Added in print. During the last revision of the paper, we realized that the proof of our main Theorem (Theorem 4.2) can be simplified thanks to a result of Bezem and Klop. Indeed item (iii) of Theorem 1.2.3 p. 18 of [22] says that an abstract rewriting system enjoying local confluence, WN and such that it is possible to define an increasing size, enjoys also SN. In our framework, this means that in order to prove $\mathrm{WN}^{\neg^{\mathsf{e}}} = \mathrm{SN}^{\neg^{\mathsf{e}}}$ (Proposition 4.10) we only need local confluence instead of confluence. This essentially means that in order to prove Theorem 4.2 one can omit Section 3. Indeed Lemmata 4.12, 4.14 and 4.16 suffice to prove the local confluence of the labelled cut-elimination. By the way, notice that this holds for the fragment MELL^2 too, so that Danos' standardization theorem (théorème 8.31 p.64 of [2]), and thus SN for MELL^2, can be proven using local confluence instead of confluence.

However, we decided to keep the present version of the paper since we believe that confluence for sps (Theorem 4.18 and Remark 4.19), so as confluence for Danos' nets , are interesting results by themselves. Furthermore, the `str`-measure defined in Section 3 is rather sharp and can be generalized to other notions of net (like for example differential nets [25], [26]); it should be a useful tool also for λ-calculus with explicit substitutions (see for example [4]).

Let us also mention that correctness is used in our proof of SN in three points: under the form of the AC condition, (i) to prove the SN of structural reduction (Proposition 3.10), (ii) to prove the local confluence of structural reduction (Lemma 4.14), hence confluence of cut-elimination (Theorem 4.18); under the form of sequentialization, (iii) to prove $\mathrm{WN}^{\neg^{\mathsf{e}}}$ (Theorem 5.11). The simplified proof just mentioned uses only (ii) and (iii).

Contents


Fact 3.6
Fact 3.9
Prop. 2.16
AC stability
Prop. 3.4
Lemma 4.12
Prop. 3.10
Lemma 4.14
Lemma 4.16
Lemma 4.3
Prop. 4.13
logℓ-confluence
Lemma 2.7
Newman
Prop. 4.15
strℓ-confluence
Lemma 4.17
commutation
logℓ/strℓ
Lemma 4.4
Th. 4.18
ℓ-confluence
Lemma 2.8
Hindley-Rosen
Lemma 2.9
Di Cosmo-Piperno-Geser
Lemma 4.9
Prop. 5.9
Prop. 4.5
SN¬e ⇔ SN
Prop. 4.10
WN¬e ⇔ SN¬e
Lemma 5.3
Lemma 5.5
Th. 5.11
WN¬e
Th. 4.2
standardization
WN¬e ⇔ SN
Prop. 5.1
Prop. 5.6
Th. 5.12
SN

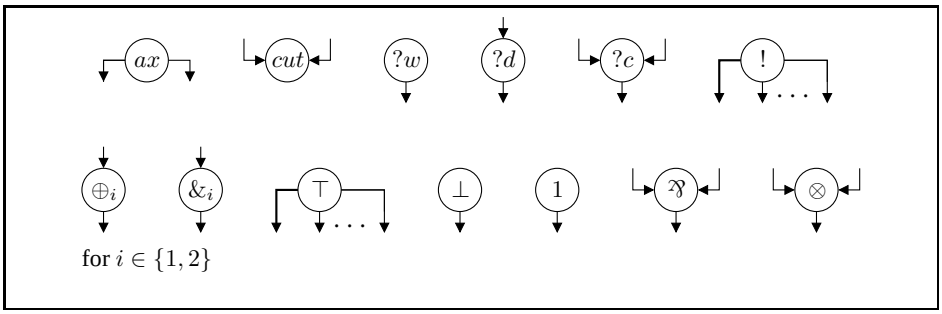


Figure 2: Sps links

2. Definitions

In this section we introduce the main tools used in the sequel of the paper. In Subsection 2.1 we define sliced pure structures (sps), Subsection 2.2 is devoted to recall some standard results in rewriting theory that will be applied to cut-elimination of sps, and in Subsection 2.3 we define cut-elimination for sps. After the main definitions, we illustrate by means of several examples the notions previously introduced (Subsection 2.4); we then conclude by presenting the AC condition (Subsection 2.5).

2.1. Sliced pure structures

We start by extending to full LL the definition of "sliced proof-structure" given in [16] for the polarized fragment. In the style of [17], we work in an untyped framework.

Definition 2.1 (Flat). A *flat* is a finite (possibly empty) labelled directed graph whose nodes (also called links) are defined together with an arity and a coarity, that is a given number of incident edges called the *premises* of the node and a given number of emergent edges called the *conclusions* of the node. The valid nodes are pictured in Figure 2.

The !-links and the $\top$-links have a distinguished conclusion (denoted in Figure 2 by a bold arrow) called *main* conclusion of the link; the other conclusions, if any, are the *auxiliary* conclusions. We allow edges with a source but no target, they are called *conclusions* of the flat.

Links will be denoted by Latin letters $l, m, \ldots$ Flats will be denoted by initial Greek letters $\alpha, \beta, \ldots$

Notice that every edge must be *conclusion of a link*, but it needs not be *conclusion of a flat*. When drawing flats we represent edges oriented downwards and we will omit to write explicitly the orientation of the edges. Moreover, we speak of moving "downwards" or "upwards" in the graph, and of nodes or edges "above" or "under" a given node/edge.

Sliced pure structures are, basically, multisets of slices, and slices are flats having the !-links parameterized by sliced pure structures. This means that slices and sliced pure structures must be defined simultaneously by induction on the exponential depth, i.e. on the number d of nesting of !-links.

Definition 2.2 (Sliced pure structure). A slice of depth at most 0 is simply a flat without !-links. A *slice of depth at most* $d+1$ is a flat α such that with every !-link o of α with n_o+1 conclusions is associated a sliced pure structure of depth at most d, called the *box of* o and denoted by π^o, with n_o *auxiliary* conclusions corresponding to the n_o auxiliary conclusions of o and another conclusion, the *main* conclusion of the box, corresponding to the main conclusion of o.

A *sliced pure structure*, sps for short, *of depth at most* d is a finite (possibly empty) multiset of slices of depth at most d with the same conclusions: we mean that an sps comes equipped with an equivalence relation on the conclusions of its slices s.t. every equivalence class contains exactly one conclusion for each slice. The conclusions of the sps are the equivalence classes of the conclusions of its slices.

We denote slices by initial Greek letters $\alpha, \beta, \ldots$, sps by final Greek letters $\pi, \sigma, \ldots$

The depth of a slice α (resp. an sps π) is the least d such that α (resp. π) is of depth at most d. A *flat* α *at depth* d in an sps π is a flat at depth d in a slice β of π; α is at depth d in β if $d=0$ and α is β (considered as a flat), or $d>0$ and α is a flat at depth $d-1$ in a box associated with a !-link of β (considered as a flat). A link l at depth d in π is a link l of a flat α at depth d in π. We denote by $\mathrm{depth}^{\pi}(l)$ the depth d of l in π. We refer more generally to a link/flat of π meaning a link/flat at some depth in π. We use the same terminology for the edges of π. We denote by $!^0(\pi)$ the set of !-links at depth 0 in π. The *size* of π, denoted $\mathrm{size}(\pi)$, is the number of links (at any depth) of π.

Given a link l of an sps π, we will often speak of "the flat of l" always meaning the flat at some depth in π containing l. The reader should notice that our sps are *multi*sets of slices, and not simply sets, as it is in [16], [17]: this quibble is needed to avoid an unnatural erasing of slices during cut-elimination. We refer to Subsection 2.4 for examples of this phenomenon. We use the additive notation for multisets: 0 is the empty multiset, $\pi+\sigma$ is the disjoint union of π and σ (repetitions do matter); an sps π can also be written as $\sum_{\alpha\in\pi}\alpha$. As a consequence, when we write $\alpha\in\pi$, we are considering an occurence of α in the multiset π, and when that expression binds an operator, as for example in $\sum_{\alpha\in\pi}$, we mean that $\sum_{\alpha\in\pi}$ varies on the set of occurences of π's slices.

Figure 3 is an example of sps of depth 1; the correspondence between the conclusions of the box of a !-link and the conclusions of the !-link is given in the figure by the order of the edges (from left to right).

Remark 2.3. Notice that, by definition, the boxes of an sps satisfy a *nesting condition*: two boxes are either disjoint or contained one in the other.

Remark 2.4. Once the decision to work without types has been taken, the question arising is: to which extent? A possibility was to use recursive types (like in [2], [20]), another one to type only ?-edges (like in [17]). In the general LL case that we consider here, none of the answers is completely satisfactory and we decided to work in a strongly untyped framework. There are little surprises following this choice: the main one is that a clash (Definition 2.10) might become reducible after some cut-elimination steps (Definition 2.12): for example, the cut s of Figure 3 is a clash, but it becomes reducible after the reduction of the cut r (see Subsection 2.4). However such oddities cause no problem w.r.t. our purposes.

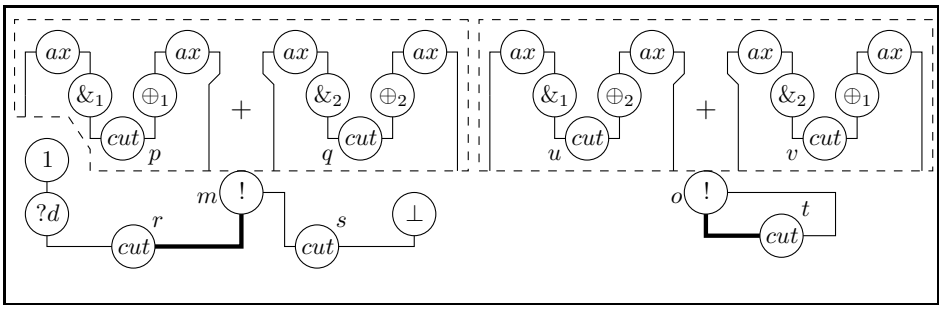


Figure 3: An example of sps

Remark 2.5. In order to have a very general result we had to use all the links of [7], including $\top$. A possible (and rather natural) choice was to rule out that link and to represent the $\top$ rule as empty sps (like in [14]). Indeed, such a choice would realize a greater quotient on proofs: this notion of sps is "closer" to denotational semantics. However, the absence of $\top$ in sps makes it more difficult to infer strong normalization of LL nets from strong normalization of sps (Proposition 5.6). More precisely, we would lose Lemma 5.3: this is the reason for our choice.

Anyway all the properties we prove for our sps still hold for $\top$-free sps (in particular Theorem 4.2).

Remark 2.6. Concerning the presence of empty structures, notice that:

- the empty flat does exist (and so do the empty slice and sps containing the empty slice), and it has no conclusion. Its presence is required by the cut-elimination procedure (Definition 2.12): the procedure applied (for example) to the sps containing a unique flat consisting of a 1-link a cut and a $\bot$-link yields the empty graph;
- with a !-link o of an sps, it is *never* possible to associate an sps containing the empty slice: o has at least one conclusion and this has also to be the case for the sps associated with o;
- empty sps (that is empty multisets of slices) do exist: there is one such sps for every set of conclusions.

The reader should notice the difference between the empty slice, that is the empty graph, and empty sps, that is empty multisets of slices. They are two different kinds of emptyness, the first one is multiplicative (it can be produced by eliminating a multiplicative cut), while the second one is additive (it can be produced by eliminating an additive cut).

2.2. Compendium of rewriting theory

Before introducing the cut-elimination on sps, we give a short reminder of some standard terminology and results in rewriting theory, which will be used in the sequel. Our references will be [12] and [22].

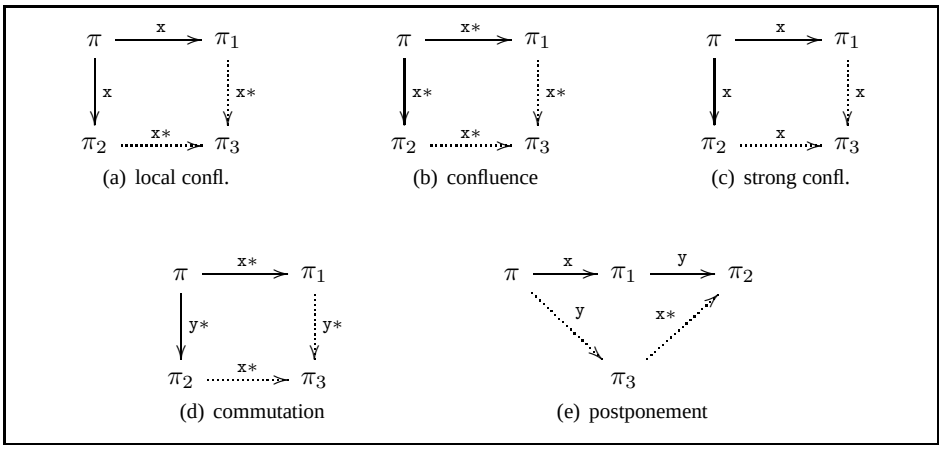


Figure 4: Diagrams for confluence, commutation and postponement

Let $\xrightarrow{\mathtt{x}}$ be a binary relation on sps, then $\xrightarrow{\mathtt{x}=}$, $\xrightarrow{\mathtt{x}+}$ and $\xrightarrow{\mathtt{x}*}$ denote, respectively, the reflexive, transitive and reflexive/transitive closure of $\xrightarrow{\mathtt{x}}$.

Let π be a sps: π is a $\mathtt{x}$-normal form whenever there is no sps π_1 with $\pi \xrightarrow{\mathtt{x}} \pi_1$; π is *weakly* $\mathtt{x}$*-normalizable* whenever there is a $\mathtt{x}$-normal form π_1 s.t. $\pi \xrightarrow{\mathtt{x}*} \pi_1$; π is *strongly* $\mathtt{x}$*-normalizable* whenever there is no infinite sequence $(\pi_i)_{i\in\mathbb{N}}$ s.t. $\pi_0 = \pi$ and $\pi_i \xrightarrow{\mathtt{x}} \pi_{i+1}$. We denote by $\mathrm{WN}^{\mathtt{x}}$ and $\mathrm{SN}^{\mathtt{x}}$ the sets of, respectively, weakly and strongly $\mathtt{x}$-normalizable sps. Clearly we have $\mathrm{SN}^{\mathtt{x}} \subseteq \mathrm{WN}^{\mathtt{x}}$. We say that $\xrightarrow{\mathtt{x}}$ is *weakly* (resp. *strongly*) *normalizing* on a set S of sps, if $\mathrm{S} \subseteq \mathrm{WN}^{\mathtt{x}}$ (resp. $\mathrm{S} \subseteq \mathrm{SN}^{\mathtt{x}}$).

The relation $\xrightarrow{\mathtt{x}}$ is *locally confluent* if for every π, π_1, π_2 s.t. $\pi_1 \xleftarrow{\mathtt{x}} \pi \xrightarrow{\mathtt{x}} \pi_2$, there is π_3 s.t. $\pi_1 \xrightarrow{\mathtt{x}*} \pi_3 \xleftarrow{\mathtt{x}*} \pi_2$ (see Figure 4(a)); $\xrightarrow{\mathtt{x}}$ is *confluent* if for every π, π_1, π_2 s.t. $\pi_1 \xleftarrow{\mathtt{x}*} \pi \xrightarrow{\mathtt{x}*} \pi_2$, there is π_3 s.t. $\pi_1 \xrightarrow{\mathtt{x}*} \pi_3 \xleftarrow{\mathtt{x}*} \pi_2$ (see Figure 4(b)); $\xrightarrow{\mathtt{x}}$ is *strongly confluent* if for every π, π_1, π_2 s.t. $\pi_1 \xleftarrow{\mathtt{x}} \pi \xrightarrow{\mathtt{x}} \pi_2$, there is π_3 s.t. $\pi_1 \xrightarrow{\mathtt{x}} \pi_3 \xleftarrow{\mathtt{x}} \pi_2$ (see Figure 4(c)). Strong confluence implies confluence and confluence implies local confluence, and none of the converse implications holds (as it is well-known).

Two relations $\xrightarrow{\mathtt{x}}$ and $\xrightarrow{\mathtt{y}}$ *commute*, if for every π, π_1, π_2 s.t. $\pi_1 \xleftarrow{\mathtt{x}*} \pi \xrightarrow{\mathtt{y}*} \pi_2$, there is π_3 s.t. $\pi_1 \xrightarrow{\mathtt{y}*} \pi_3 \xleftarrow{\mathtt{x}*} \pi_2$ (see Figure 4(d)); the relation $\xrightarrow{\mathtt{x}}$ *can be postponed* w.r.t. the relation $\xrightarrow{\mathtt{y}}$, when for every π, π_1, π_2 s.t. $\pi \xrightarrow{\mathtt{x}} \pi_1 \xrightarrow{\mathtt{y}} \pi_2$, there is π_3 s.t. $\pi \xrightarrow{\mathtt{y}} \pi_3 \xrightarrow{\mathtt{x}*} \pi_2$ (see Figure 4(e)).

Lemma 2.7 (Newman). *A locally confluent and strongly normalizing relation is confluent.*

Lemma 2.8 (Hindley-Rosen). *If two relations $\xrightarrow{\mathtt{x}}$, $\xrightarrow{\mathtt{y}}$ are confluent and commute, then their union $\xrightarrow{\mathtt{x}} \cup \xrightarrow{\mathtt{y}}$ is confluent.*

Lemma 2.9 (Di Cosmo-Piperno-Geser). *Let $\xrightarrow{\mathtt{x}}$, $\xrightarrow{\mathtt{y}}$ be two relations s.t. $\xrightarrow{\mathtt{x}}$ is strongly normalizing and for every π, π_1, π_2 s.t. $\pi_1 \xleftarrow{\mathtt{y}} \pi \xrightarrow{\mathtt{x}} \pi_2$, there is π_3 s.t. $\pi_1 \xrightarrow{\mathtt{x}+} \pi_3$ and*

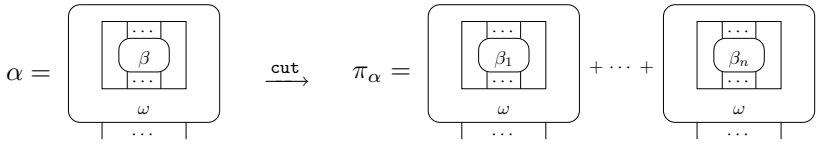
$\pi_2 \xrightarrow{\mathtt{y}^*} \pi_3$, *that is the following diagram holds*

$$\begin{array}{ccc} \pi & \xrightarrow{\mathtt{x}} & \pi_2 \\ {\scriptstyle \mathtt{y}}\downarrow & & \downarrow{\scriptstyle \mathtt{y}^*} \\ \pi_1 & \xrightarrow{\mathtt{x}+} & \pi_3 \end{array} \tag{1}$$

then $\xrightarrow{\mathtt{x}}$ *and* $\xrightarrow{\mathtt{y}}$ *commute.*

2.3. Cut-elimination

We now define the binary relation of cut-elimination $\xrightarrow{\mathtt{cut}}$ on sps. Before the details let us give an idea of how $\xrightarrow{\mathtt{cut}}$ works: to eliminate a cut at depth 0 in a slice α means in general to transform α in an sps $\pi_\alpha = \sum_{i\in I} \alpha'_i$, s.t. every slice α'_i is obtained from α by substituting a specific subgraph β of α with a subgraph β_i having the same pending edges (i.e. edges with no target or no source) as β, as pictured below:



The number n and the subgraphs $\beta, \beta_1, \ldots, \beta_n$ depend on the cut we want to reduce. Definition 2.11 formalizes the notion of substituting a subgraph in a sps introducing the notion of module and one-hole context; Definition 2.12 and Figure 5 give the substitutions which define the cut-elimination.

As mentioned in the introduction, our sps are very general structures: they are untyped and they may be incorrect (w.r.t. the proofs of LL). Even in such a general setting cut-elimination can be defined and nice properties hold, as we will show in the sequel of the paper. However, one has to handle carefully some strange phenomenons related to this rather "wild" situation, like the presence of cuts which are not reducible:

Definition 2.10 (Clash and deadlock). The two edges premises of a cut link are *dual* when one of the following conditions holds:

- they are conclusions of a $\otimes$-node and of a $\parr$-node,
- they are conclusions of a $\oplus_i$-node and of a $\&_j$-node (for $i, j \in \{1, 2\}$),
- they are conclusions of a 1-node and of a $\bot$-node,
- one is the main conclusion of a !-node and the other one is either an auxiliary conclusion of a !-node, or the conclusion of one of the following nodes: $?c$, $?w$, $?d$.

A cut node of an sps is:

- a *clash*, when the premises of the cut node are not dual edges and none of the two is the conclusion of an ax-link nor an auxiliary conclusion of a $\top$-link;
- a *deadlock*, when the two premises of the cut link are conclusions of the same ax-link (resp. !-link, $\top$-link);
- *reducible*, otherwise.

In Figure 3, for example, t is a deadlock, s is a clash and the five other cuts are reducible.

Definition 2.11 (Module, context closure, substitution). A *module* μ is an sps whose slices at depth 0 may have edges without source, called *hypotheses*. In addition to having the same conclusions, the slices of a module are required to have the same hypotheses, i.e. a module comes equipped with an equivalence relation on the hypotheses of its slices at depth 0 s.t. every equivalence class contains exactly one hypothesis for each slice at depth 0[8].

A *one-hole context* $\omega[\,]$ is an sps having exactly one occurrence of a special cell, the *hole*, which has an arbitrary arity and coarity. This formally means that $\omega[\,] = \sigma + \alpha[\,]$, where σ is an sps and $\alpha[\,]$ is a slice having at depth 0 either exactly one occurrence of the hole or a !-link o such that with o is associated a one-hole context $\rho[\,]$.

Given a one-hole context $\omega[\,] = \sigma + \alpha[\,]$ and a module μ equipped with a bijection between the hypotheses (resp. conclusions) of μ and the premises (resp. conclusions) of the hole in $\omega[\,]$, we define the sps $\omega[\mu]$ by induction on the depth of the hole in $\omega[\,]$:

- if the hole has depth 0 in $\alpha[\,]$, then $\omega[\mu] = \sigma + \sum_{\beta \in \mu} \alpha[\beta]$, where $\alpha[\beta]$ is the slice obtained by substituting in $\alpha[\,]$ the slice β for the hole, i.e. identifying the hypotheses (resp. conclusions) of β with the corresponding premises (resp. conclusions) of the hole;
- if the hole is a cell of the one-hole context $\rho[\,]$ associated with a !-link o at depth 0 in α, then $\omega[\mu]$ is obtained by associating with o the sps $\rho[\mu]$.

The *context closure* of a binary relation R between modules is the smallest relation containing R and such that for every one-hole context $\omega[\,]$, $\mu \mathrm{R} \mu'$ implies $\omega[\mu] \mathrm{R} \omega[\mu']$. We also say that $\omega[\mu']$ is the result of *substituting* μ' *for* μ in $\omega[\mu]$.

Definition 2.12 (Types of cut and cut-elimination). We define the *reduction steps* as the following relations between a single slice module β, the *redex*, containing a reducible cut t and a module $\sum_i \beta_i$, the *contractum*. Apart from the cases $(\&_i/\oplus_j)$, $(!/?d)$, the contractum is a single slice β', too. All reduction steps are pictured in Figure 5.

(ax)**:** β is made of t and two distinct links l,n, each of them having one premise of t as conclusion; moreover, one link between l,n, say l, is an ax-link and if n is a $\top$-link, then the edge shared by n and t is the main conclusion of n. In this case the contractum β' is simply the link n.

[8] Of course since the slices at depth 0 of a module have the same conclusions, there is also an equivalence relation on the conclusions of the slices at depth 0 of a module, as explained in Definition 2.2.

$(1/\bot)$**:** β is made of t, a 1-link and a $\bot$-link; a premise of t is the conclusion of the 1-link and the other is the conclusion of the $\bot$-link. In this case the contractum β' is the empty graph.[9]

$(\otimes/\parr)$**:** β is made of t, a $\parr$-link and a $\otimes$-link; a premise of t is the conclusion of the $\otimes$-link and the other is the conclusion of the $\parr$-link. In this case the contractum β' is obtained from β by erasing the $\parr$-link, the $\otimes$-link and the cut link t (and its premises) and by putting two new cut links between the two left (resp. right) premises of the $\parr$-link and of the $\otimes$-link[10].

$(\top/\mathrm{cc})$**:** β is made of t, a $\top$-link l having one premise of t among its auxiliary conclusions and a slice γ (not containing l) having the other premise of t among its conclusions. Let us call a (resp. b) the edge shared by t and l (resp. t and γ). The contractum β' is a $\top$-link, which we still call l, with conclusions the conclusions of l different from a and the conclusions of γ different from b.

$(\oplus_i/\&_j)$**:** β is made of t, a $\oplus_i$-link, and a $\&_j$-link; a premise of t is the conclusion of the $\oplus_i$-link and the other is the conclusion of the $\&_j$-link. If $i = j$, then the contractum β' is obtained from β by erasing the two links (and their conclusions) and by moving up the cut link to their premises. If $i \neq j$, then the contractum is the empty multiset.

$(!/?d)$**:** β is made of t, a !-link with main conclusion a premise of t and a $?d$-link with conclusion the other premise of t. Let ρ be the sps associated with the !-link. If ρ is the empty multiset, then so it is the contractum. Otherwise, with each slice γ of ρ, we associate the slice γ' defined by cutting γ's main conclusion[11] with the premise of the $?d$-link: the contractum is then $\sum_{\gamma\in\rho}\gamma'$.

$(!/?w)$**:** β is made of t, a !-link with main conclusion a premise of t and a $?w$-link with conclusion the other premise of t. In this case, the contractum β' is made of as many $?w$-links as the auxiliary conclusions of the !-link[12].

$(!/?c)$**:** β is made of t, a !-link l with main conclusion a premise of t and a $?c$-link n with conclusion the other premise of t. In this case, the contractum β' is obtained from β as follows: let's call a_1 and a_2 the two edges premises of the $?c$-link n. We create a new !-link l' by copying the link l, and we pairwise contract the auxiliary conclusions of l and l': the conclusions of these new $?c$-links substitute the auxiliary conclusions of l in β. We then erase n (and its conclusion) and t, and we connect the main conclusion of l (resp. l') with a_1 (resp. a_2) by means of a cut link. The sps associated with l and l' are the same.

---

[9] This case yields the *singleton* of the empty graph, that has not to be confused with the *empty multiset*, contractum of the $(\oplus_i/\&_j)$ redex.

[10] Notice that this means that the premises of the $\otimes/\parr$-links are *ordered*; we shall see in the transformation associated with the $(!/?c)$ cut link that this is not the case of the premises of the $?c$-links (nor of the premises of the cut links).

[11] We extend here the notion of "main conclusion of a box" to every slice of the box.

[12] In case the !-link has no auxiliary conclusion, the contractum is the singleton of the empty graph, like in the $(1/\bot)$ case.

$(!/!)$: β is made of t, a !-link l with main conclusion a premise of t and a !-link l' having the other premise of t among its auxiliary conclusions. Let us call a' the edge shared by l' and t, and ρ the sps associated with l'. The contractum β' is a !-link, which we still denote by l', having as conclusions the conclusions of l' different from a' and the auxiliary conclusions of l, and having as box the sps $\sum_{\gamma\in\rho}\gamma'$, where γ' is obtained by cutting the auxiliary γ's conclusion corresponding to a' with the main conclusion of l (the sps associated with l remains unchanged)[13].

The *cut-elimination* $\xrightarrow{\mathtt{cut}}$ is the context closure of the union of the reduction steps. Given a sps π and a reducible cut t in π, we denote by $t(\pi)$ the sps[14] obtained by replacing the redex associated with t by its contractum. We will also refer to $t(\pi)$ as a one step reduct of π. We say that $\mathtt{x}$ is the type of t whenever $\pi \xrightarrow{\mathtt{x}} t(\pi)$. In the sequel we will denote the set $\mathrm{WN}^{\mathtt{cut}}$ (resp. $\mathrm{SN}^{\mathtt{cut}}$) of weakly (resp. strongly) normalizable sps w.r.t. $\xrightarrow{\mathtt{cut}}$ simply by WN (resp. SN).

Remark 2.13.

- Notice that the cut-elimination procedure is defined without any reference to correctness.
- Observe the restriction imposed in the (ax) case: in order for a cut to be of type (ax), not only one premise of t must be conclusion of an axiom, but also the other premise cannot be an auxiliary conclusion of a $\top$-link. This restriction makes every cut link of an sps of a unique type: in the absence of it, there would be a (unique) case in which a reducible cut link t of an sps might have two different types ((ax) and $(\top/\mathrm{cc})$). Notice that this little problem would not occur if $\top$ were rejected from sps links (see Remark 2.5). Anyway, this constraint does not restrict the possible reductions starting from a sps, because there is an obvious $(\top/\mathrm{cc})$ reduction step having exactly the same effect as the (ax) reduction step: choose the axiom link as the γ in the redex of $(\top/\mathrm{cc})$ (see Figure 5). Working with reducible cut links having a unique type is simpler and useful in Section 4, when we define the notion of erasing cut (Definition 4.1).
- Notice that the $(\top/\mathrm{cc})$ step gives rise to non deterministic (and non confluent) reductions: for example, a cut whose premises are auxiliary conclusions of two distinct $\top$-links l, l' can be reduced by erasing either l or l'. Anyway such phenomena disappear when one considers only non erasing reductions (see Definition 4.1).

We now give a precise definition of the notions of ancestor and residue of a node: the point is to know whether a node of $t(\pi)$ has been created by the cut-elimination procedure or was already a node of π.

[13] Notice that if ρ is the empty multiset, so is the box associated with l'.
[14] The fact that $t(\pi)$ is indeed an sps can be easlily checked.

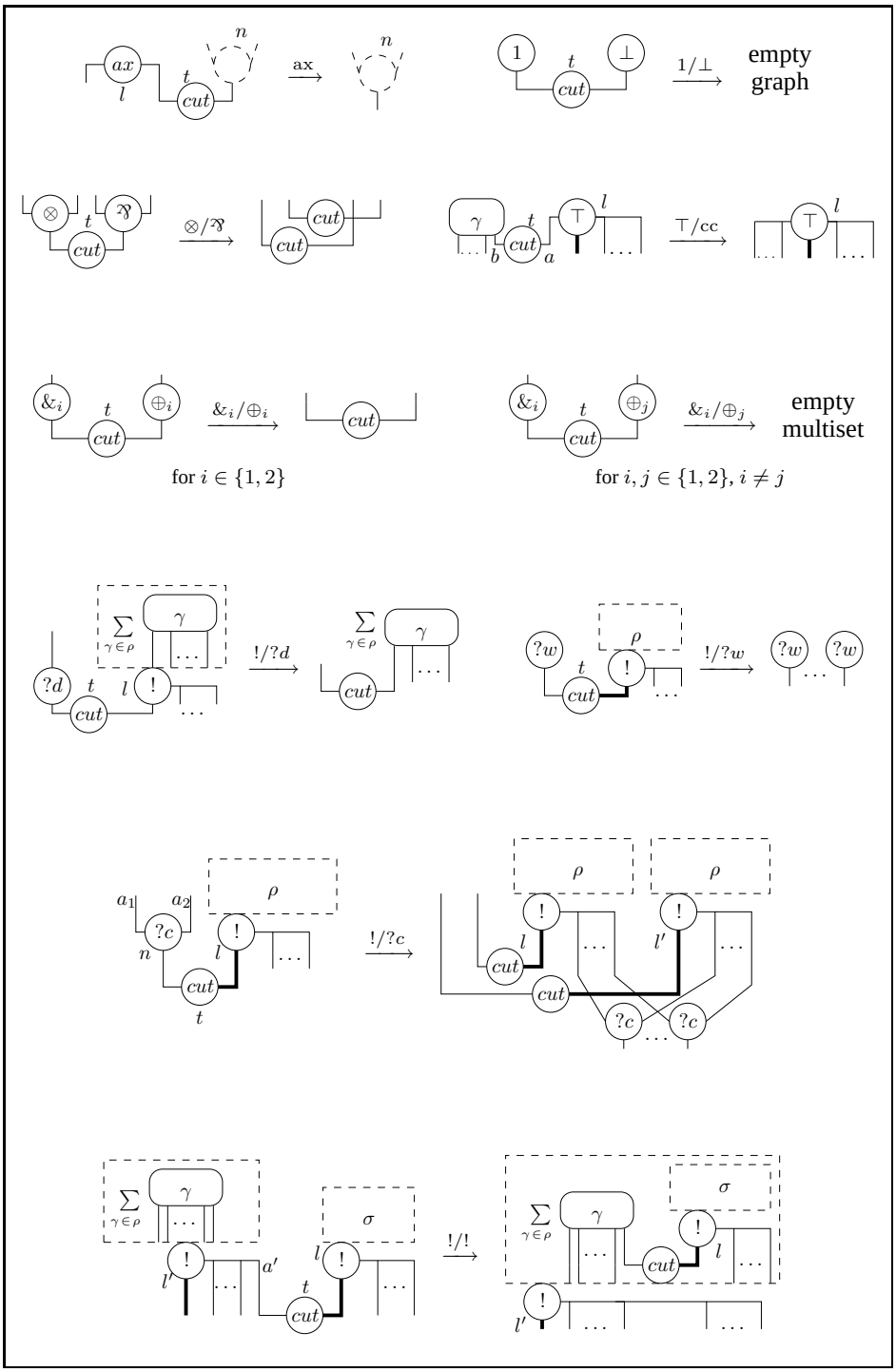


Figure 5: The reduction steps for cut-elimination

Definition 2.14 (Ancestor, residue). Let π be an sps, t be a cut link of π and $t(\pi)$ be a one step reduct of π associated with t. When a node l of $t(\pi)$ comes from a (unique) node $\overleftarrow{l}$ of π, we say that $\overleftarrow{l}$ is the *ancestor* of l in π and that l is a *residue* of $\overleftarrow{l}$ in $t(\pi)$. If this is not the case, then l has no ancestor in π, and we say it is a *created* node. We indicate, for every type of cut node t of Definition 2.12, which links are created in $t(\pi)$ (meaning that the other nodes of $t(\pi)$ are residues of some π's node). We use the notations of Definition 2.12 and Figure 5:

(ax)**:** there are no created nodes in $t(\pi)$;

$(1/\bot)$**:** there are no created nodes in $t(\pi)$;

$(\otimes/\parr)$**:** the two new cut links between the two left (resp. right) premises of the $\parr$-link and of the $\otimes$-link are created nodes;

$(\top/\mathrm{cc})$**:** there are no created nodes in $t(\pi)$;

$(\oplus_i/\&_j)$**:** if $i = j$, then the cut link between the two premises of the $\oplus_i/\&_i$-links is a created node. If $i \neq j$, there are no created nodes in $t(\pi)$;

$(!/?d)$**:** every cut link between γ_i's main conclusion and the premise of the $?d$-link is a created node;

$(!/?w)$**:** all the $?w$-links added during this step are created links;

$(!/?c)$**:** the new $?c$-links having as premises the auxiliary conclusions of l and l' are created nodes. The two cut links having among their premises the main conclusions of l and l' are created nodes;[15]

$(!/!)$**:** every cut link between the auxiliary γ's conclusion corresponding to a' and l's main conclusion is a created link. (Notice that the "new" !-link l' is considered a residue of the corresponding !-link of π, even though it might have different conclusions).

2.4. Examples of reductions

After so many definitions, some examples might be useful.... Let us apply cut-elimination to the sps π of Figure 3. If one reduces the $(\&_1/\oplus_1)$ cut p, the $(\&_2/\oplus_2)$ cut q, and then the two created cuts of type ax, one obtains the sps π' of Figure 6. Notice that the sps π^m associated with the !-link m has now two occurences of the same slice (consisting of an ax-link): if we had defined the sps as *sets* of slices, we would have missed one occurence of the slice in π^m, so giving an "erasing" feature to the $(\&_i/\oplus_i)$ step which is quite unnatural. As already mentioned in the Introduction, it is crucial in order to apply Gandy's method to have a good notion of *erasing* cut-elimination step: this will appear clear in Section 4, where we split $\xrightarrow{\mathtt{cut}}$ into the erasing and the non-easing reduction (see Definition 4.1).

[15] Notice that l and l' are both residues of l.

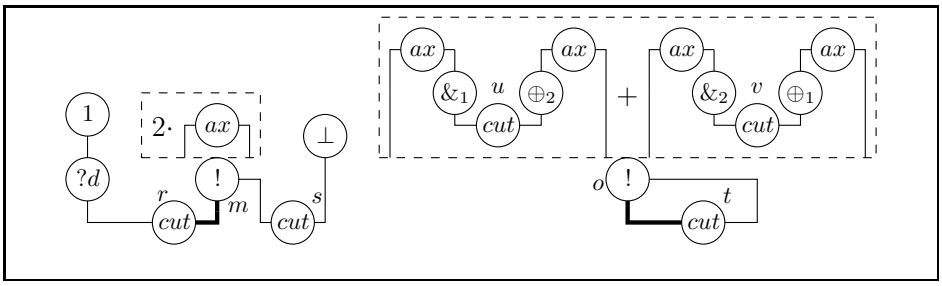


Figure 6: The result of applying cut-elimination to the sps of Figure 3

Let us go on in the elimination of the cuts in the sps π' of Figure 6. If one reduces the $(!/?d)$ cut r, then one gets the sps π'' of Figure 7. Notice that this reduction duplicates the unique slice of π', since it opens a box containing two slices. This is a tricky feature of sps (due to the presence of additive slices): the reduction of a cut of type $(!/?d)$ (like r) may duplicate other cuts at the same exponential depth (like the cuts t and s in π'). We tame this kind of duplication in Section 3, where we prove that the "logical" subreduction of $\xrightarrow{\texttt{cut}}$ (Definition 3.1) is strongly normalizing (Proposition 3.4).

Yet another remark on the reduction of r: notice that the residues of the clash s of Figure 6 are reducible cuts of Figure 7. Thus, let us reduce these residues of s, the $(1/\bot)$ cuts so obtained and the two $(\&_i/\oplus_j)$ $(i \neq j)$ cuts u and v in the box associated with o: we obtain the sps π''' of Figure 8. Notice that the two reduction steps of type $(\&_i/\oplus_j)$ $(i \neq j)$ have erased both the slices of the sps associated with o, transforming it in the empty multiset of slices. This is really different from the reduction of the cut of type $(1/\bot)$, which has transformed the subgraph consisting in the 1-link, the $\bot$-link and the cut in the empty graph: pay attention not to confuse empty sps (i.e. empty multiset of slices) with the empty slice (see also Remark 2.6). The sps π''' of Figure 8 is a `cut`-normal form. Notice however that π''' contains the deadlock t: in general `cut`-normal forms may contain non-reducible cuts, i.e. clashes or deadlocks.

Let us come now to the problem of normalization. The cut-elimination procedure applied to an sps π may lead to infinite reduction sequences basically in two distinct cases: (i) either because π is not typable (by LL formulas, see the grammar in Fig-

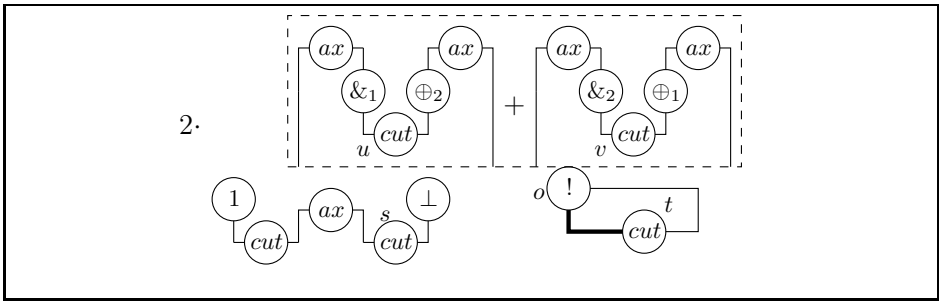


Figure 7: The result of applying cut-elimination to the sps of Figure 6

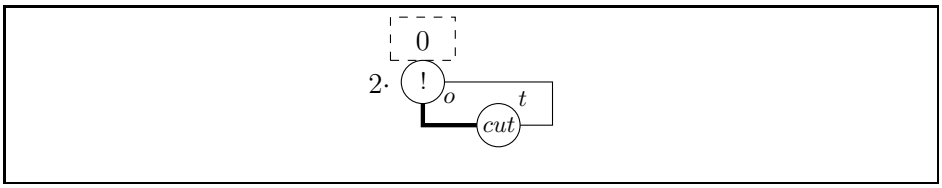


Figure 8: The result of applying cut-elimination to the sps of Figure 7

ure 17) or (ii) because π is not correct (w.r.t. a notion of correctness which will be introduced in the following Subsection 2.5). We give an example of (i) in Figure 9, and examples of (ii) are in Figure 11 and in Figure 12. Let us comment a bit each of them.

The sps $\delta\delta$ of Figure 9 is taken from [18]: it is a simplification in the setting of nets of the most famous λ-term which is not normalizable, $\Delta\Delta$, with $\Delta = \lambda x.xx$. Figure 10 shows that $\delta\delta$ reduces to itself. The sps $\delta\delta$ is not even weakly normalizable, and it is not typable by LL formulas, even if it is correct in the sense that it satisfies the AC condition of Definition 2.15.

Let us now consider the slice α of Figure 11: α is typable by LL formulas, but it is not switching acyclic (Definition 2.15) owing to the cycle crossing the cut, the !-link and the $?c$-link. Such a cycle is the actual reason for the loop pictured in Figure 11.

We can use the slice α of Figure 11 to show a last intriguing example of cut-elimination. Let γ be the slice obtained from α by erasing the cut link, then consider the slice β defined in Figure 12: β is a counter-example both to the confluence and to the normalization of cut-elimination for sps which are not correct (they don't satisfy the AC condition of Definition 2.15). On the one hand, if one reduces the $(!/!)$ cut t, then the created cut $(!/?d)$ and last the created $(\otimes/\parr)$ cut, one obtains the slice β_1 of Figure 12, which is not weakly normalizable since reducing the cut t' leads to a looping cut-elimination, similar to the one described in Figure 11. On the other hand, if one reduces the cut u in β, then one obtains the slice β_2 of Figure 12, which is even strongly normalizable: its (unique) normal form is β_3 (notice that the cut t has become a deadlock, so it is not reducible any more). This example clearly shows that if we drop the AC condition, we loose both confluence and weak (thus strong) normalization *even*

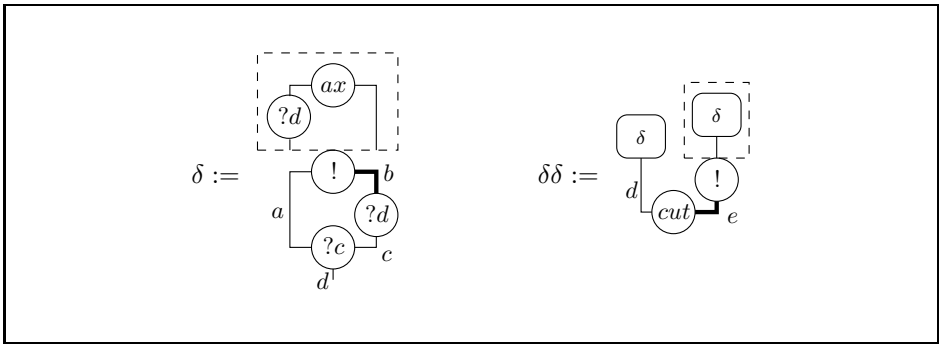


Figure 9: The slice $\delta\delta$ which is not WN

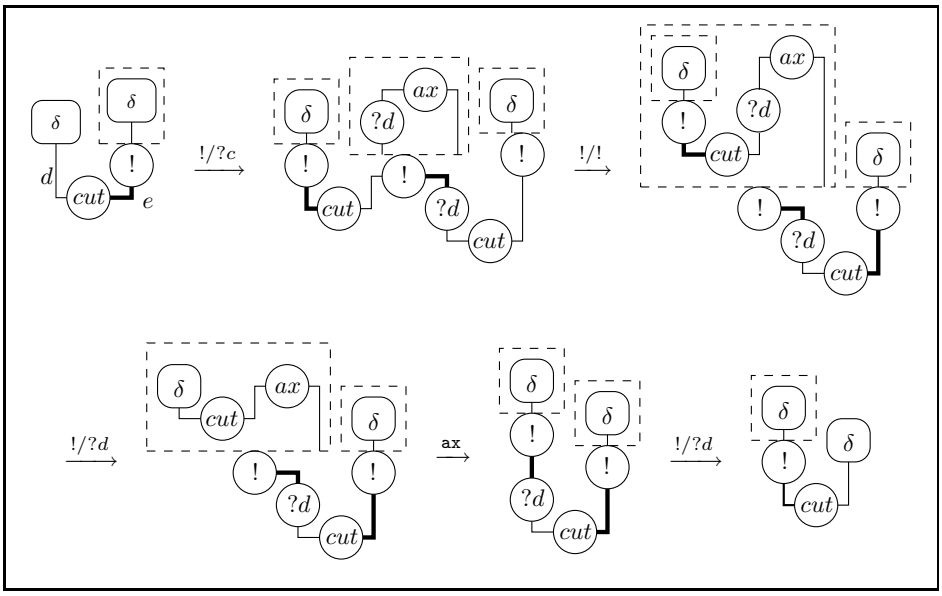


Figure 10: A proof that $\delta\delta \xrightarrow{\texttt{cut+}} \delta\delta$

for typable sps.

The main result of this paper is the standardization theorem for correct sps (i.e. for sps satisfying AC, Definition 2.15): for correct sps $\mathrm{WN}^{\neg^{\mathsf{e}}}$ implies SN (Theorem 4.2). One crucial step in the proof of Theorem 4.2 is Proposition 4.10: for correct sps non-erasing weak normalization coincides with non-erasing strong normalization. Observe that the slice β of Figure 12 gives a counter-example to this equivalence for sps which are not correct (which don't satisfy the AC condition of Definition 2.15): β can be normalized without applying erasing steps (in the precise sense of Definition 4.1), but it is not strongly normalizable. More in detail, this counter-example is due to (i) the presence of deadlocks (the cut t is not erased by one reduction step of β, but it becomes

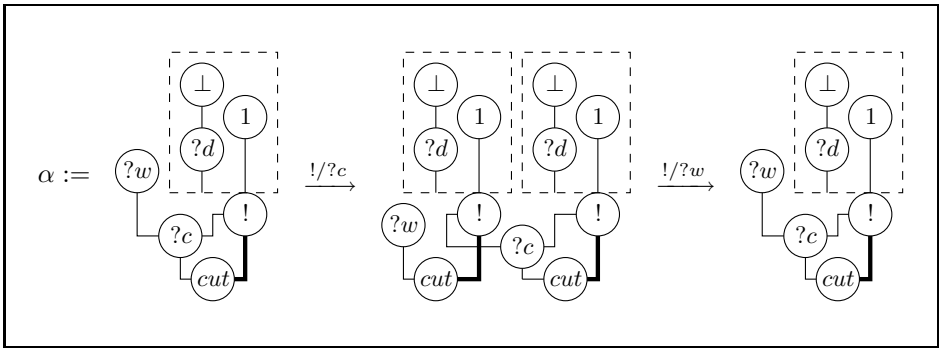


Figure 11: An example of slice α s.t. $\alpha \xrightarrow{\texttt{cut+}} \alpha$. The slice α is typable by LL formulas

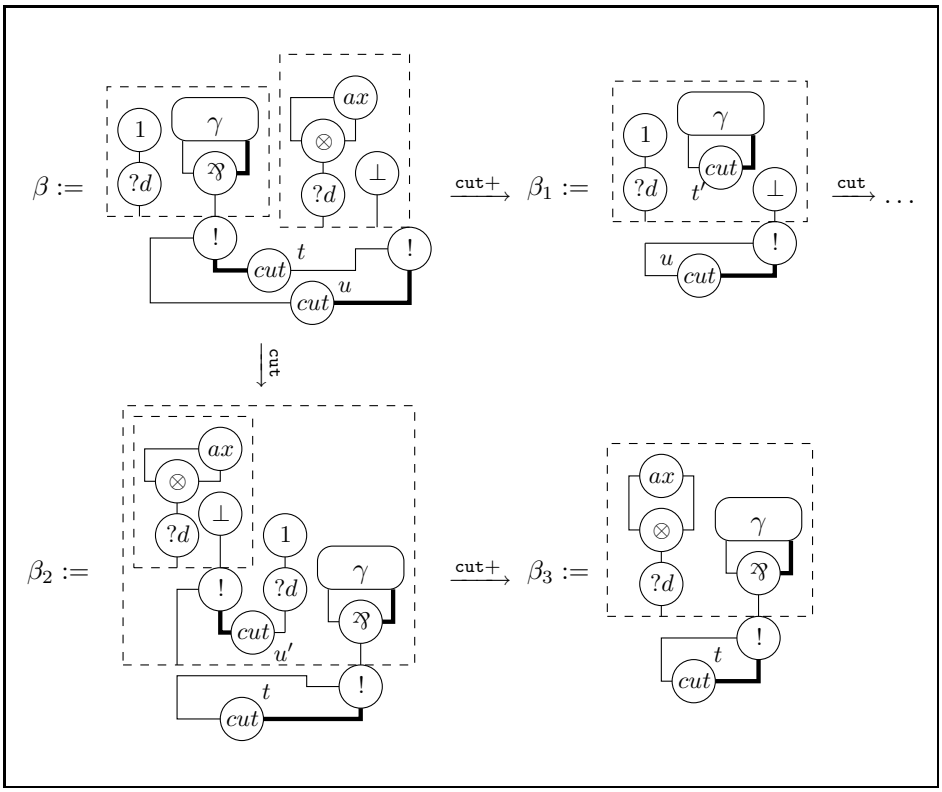


Figure 12: A counter-example to normalization and confluence of cut-elimination. The slice γ is obtained by removing the cut from the slice α pictured in Figure 11

a deadlock), (ii) the failure of the confluence of $\xrightarrow{\texttt{cut}}$ for general sps (indeed one of the main ingredients in the proof of Proposition 4.10 is the confluence of (a labelled variant of) cut-elimination on correct sps, Theorem 4.18).

2.5. Switching acyclicity

Cut-elimination can be performed without any reference to correctness. However we noticed in the previous subsection that in presence of cyclic sps there are "bad" computations (even without additives and even for typable sps): the weak (and the strong) normalization property fails, so as the confluence property (recall the example of Figure 12).

We will then use in this paper the weakest (standard) notion of correctness known in the litterature, requiring to our sps to be "switching acyclic". Switching acyclicity is required "slice by slice" (Definition 2.15). We use this condition to avoid cyclic sps (not enjoying WN), and to prove Theorem 4.18.

Definition 2.15 (AC **condition).** A *switching* of a flat α is an undirected subgraph of α obtained by forgetting the orientation of α's edges and by deleting one of the two premises of each $\parr$-node and $?c$-node of α.

We say that an sps π is *switching acyclic*, or, equivalently, that π *satisfies* AC, if every switching of every flat of π is an acyclic graph.

Examples of sps satisfying AC are the sps of Figures 9, 10 and all sps pictured in Section 5.

The following proposition is an important property of sps, which will be used in this paper to prove SN and confluence of the $\mathtt{str}$-reduction (Prop. 3.10 and Prop. 4.15):

Proposition 2.16. *Let π be an sps and t be a cut link of π which is not of type $(\top/\mathrm{cc})$*[16]*. If π satisfies AC then $t(\pi)$ satisfies AC.*

PROOF. Standard (see [2]). □

[16]One has to refuse $(\top/\mathrm{cc})$ steps, essentially because they are not "local"; for the same reason $(\top/\mathrm{cc})$ and (ccad) steps of [7] and [24] can be performed only in presence of correctness, and of a much stronger notion of correctness then AC.

3. Two results of strong normalization

In this section we define two notable subreductions of $\xrightarrow{\mathtt{cut}}$: the *logical reduction* $\xrightarrow{\mathtt{log}}$ and the *structural reduction* $\xrightarrow{\mathtt{str}}$ (Definition 3.1). The union of the logical and structural reductions is $\xrightarrow{\mathtt{cut}}$. A basic fact, crucial in the next Section 4, is that both $\xrightarrow{\mathtt{log}}$ and $\xrightarrow{\mathtt{str}}$ are SN on sps satisfying AC (Proposition 3.4 and Proposition 3.10), even if their union $\xrightarrow{\mathtt{cut}}$ is not even WN (see the example of Fig. 10). More precisely, we define two measures on sps, $|\pi|_{\mathtt{log}}$ and $|\pi|_{\mathtt{str}}$ (Definition 3.3 and Definition 3.8), and we prove that $|\pi|_{\mathtt{log}}$ (resp. $|\pi|_{\mathtt{str}}$) shrinks after every logical (resp. structural) reduction step. In addition, we briefly discuss how the structural reduction increases $|\pi|_{\mathtt{log}}$ and conversely how the logical reduction increases $|\pi|_{\mathtt{str}}$, in accordance with the fact that cut-elimination is not WN.

Let us stress that the `str`-measure is really sharp and it can be generalized to other notions of net (like for example differential nets [25], [26]).

Definition 3.1. The *logical reduction*, denoted by $\xrightarrow{\mathtt{log}}$, is the context closure of the reduction steps (ax), $(1/\bot)$, $(\otimes/\parr)$, $(\top/\mathrm{cc})$, $(\oplus_i/\&_j)$, and $(!/?d)$. The *structural reduction*, denoted by $\xrightarrow{\mathtt{str}}$, is the context closure of the reduction steps $(!/?w)$, $(!/?c)$ and $(!/!)$.

Notice that $\xrightarrow{\mathtt{log}}$ and $\xrightarrow{\mathtt{str}}$ have no reduction step in common and their union is $\xrightarrow{\mathtt{cut}}$.

3.1. The strong normalization of $\xrightarrow{\mathtt{log}}$

We now prove that $\xrightarrow{\mathtt{log}}$ enjoys strong normalization: indeed $\mathrm{SN}^{\mathtt{log}}$ contains *every* sps, regardless its correctness. This is a sharp difference with the case of $\xrightarrow{\mathtt{str}}$, where strong `str`-normalization holds only for sps satisfying AC: for example the slice α of Figure 11 does not satisfy AC, and it is not strongly `str`-normalizable (actually not even weakly `str`-normalizable).

Consider an sps π which has at most one slice, and recursively s.t. every box of π has at most one slice. For such a π, the proof that $\xrightarrow{\mathtt{log}}$ is SN is immediate: every reduction step of $\xrightarrow{\mathtt{log}}$ strictly decreases the number of links of π and keeps the property of having at most one slice in each box. However, in the general case the links of an sps might increase after a $\xrightarrow{\mathtt{log}}$ reduction: if π has a box π^o associated with a !-link o and containing more than one slice, then a $(!/?d)$ step "opening" o will (additively) duplicate the links at the same depth as o, as many times as the number of slices of π^o. This means we need to find a sharper measure on sps than their sizes, in order to point out what is decreasing under $\xrightarrow{\mathtt{log}}$. In some sense, the previous remark concerning a very special case of sps can also be used in the general case, thanks to the notion of *single-threaded slice* introduced in [16].

Definition 3.2 (Single-threaded slice, [16]). [17] A *single-threaded slice*, sgth for short, of depth at most d is a slice α of depth at most d s.t. with every !-link at depth 0 of α is

[17] There are actually two differences w.r.t. [16]: in that paper the authors define the *set* of sgth of a proof-

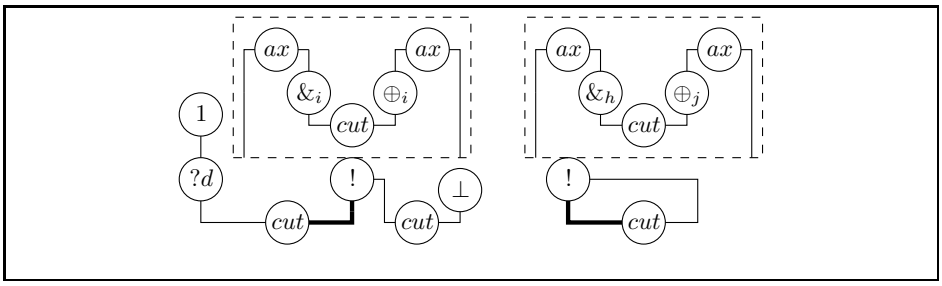


Figure 13: The multiset $\mathrm{sgth}(\pi)$, where π is the sps of Fig. 3, consists of four single-threaded slices, given by the above picture taking $i, h, j \in \{1, 2\}$, $h \neq j$.

associated either the empty sps (with the appropriate conclusion) or a single-threaded slice of depth at most $d - 1$.

Given an sps π we define $\mathrm{sgth}(\pi)$, the *multiset of* sgth *of* π, by induction on the depth of π. In case π has depth 0, then $\mathrm{sgth}(\pi) = [\pi]$; in case π has depth $d + 1$, then $\alpha \in \mathrm{sgth}(\pi)$ iff α is obtained from a slice α' of π by choosing for every !-link o at depth 0 of α':

- the empty sps (with the appropriate conclusion), in case this sps is associated with o in π,
- a sgth of the sps associated with o, otherwise.

For an example see Figure 13, which shows the sgth of the sps π of Figure 3.

Definition 3.3 (`log`-measure). The `log`-measure of an sps π is a natural number, denoted by $|\pi|_{\mathtt{log}}$ and defined as follows:

$$|\pi|_{\mathtt{log}} \quad := \quad \sum_{\alpha \in \mathrm{sgth}(\pi)} \mathrm{size}(\alpha)$$

For example, consider the sps π of Figure 3: it has four sgth and each of them has 18 links (see Figure 13), so $|\pi|_{\mathtt{log}} = 18 \times 4 = 72$. Notice that for every sps π, the `log`-measure of π is the sum of the `log`-measures of the slices of π, i.e. $|\pi|_{\mathtt{log}} = \sum_{\alpha \in \pi} |\alpha|_{\mathtt{log}}$.

Proposition 3.4. *The reduction* $\xrightarrow{\mathtt{log}}$ *is* SN *on the sps.*

PROOF. We prove that $\pi \xrightarrow{\mathtt{log}} \overline{\pi}$ implies $|\pi|_{\mathtt{log}} > |\overline{\pi}|_{\mathtt{log}}$. First, let us restrict to the case π has exactly one slice α, from which the general case follows, as showed at the end of the proof.

net, we are instead interested in the *multiset* of sgth of an sps (this difference is due to the fact that our sps are multisets of slices and not simply sets, as it is in [16]); the other difference is that while in [16] there is *exactly* one slice associated with every !-link, we have here *at most* one slice (this difference is necessary to deal with the case of !-links having an empty sps inside in Proposition 3.4).

Let $\alpha \xrightarrow{\mathtt{log}} \pi_\alpha$, we prove $|\alpha|_{\mathtt{log}} > |\pi_\alpha|_{\mathtt{log}}$. The proof splits in the six cases associated with the types of the $\xrightarrow{\mathtt{log}}$ steps: we treat in detail only the case of a $(!/?d)$ cut at depth 0 in α, the other cases being immediate. Remember Figure 5 and the notation of Definition 2.12 in the $(!/?d)$ step. Let $\Sigma_{i\leq n}\gamma_i$ ($n \geq 0$) be the sps associated with the !-link l of α, so that $\pi_\alpha = \Sigma_{i\leq n}\alpha'_i$. If $n = 0$, then π_α is empty (α is erased) and clearly $|\alpha|_{\mathtt{log}} > |\pi_\alpha|_{\mathtt{log}}$[18]. Otherwise, let $i \in \{1, \ldots, n\}$ and let $\mathrm{sgth}(\alpha)_i$ be the multiset of sgth of α choosing the slice γ_i for l (thus $\mathrm{sgth}(\alpha)_i \subseteq \mathrm{sgth}(\alpha)$). Notice that $\mathrm{sgth}(\alpha) = \sum_{i\leq n} \mathrm{sgth}(\alpha)_i$, so in particular $|\alpha|_{\mathtt{log}} = \sum_{i\leq n} |\,\mathrm{sgth}(\alpha)_i|_{\mathtt{log}}$; moreover to every i and every $\beta \in \mathrm{sgth}(\alpha)_i$ it corresponds exaclty one element $\beta' \in \mathrm{sgth}(\alpha'_i)$ and $\mathrm{size}(\beta) = \mathrm{size}(\beta') + 2$ (in β' the !-link l and the $?d$-link above t have been erased), so $|\,\mathrm{sgth}(\alpha)_i|_{\mathtt{log}} > |\alpha'_i|_{\mathtt{log}}$. Since $|\pi_\alpha|_{\mathtt{log}} = \sum_{i\leq n} |\alpha'_i|_{\mathtt{log}}$, we conclude $|\alpha|_{\mathtt{log}} > |\pi_\alpha|_{\mathtt{log}}$.

Let's consider now the general case of an sps π: suppose $\pi \xrightarrow{\mathtt{log}} \overline{\pi}$, then there is a slice $\alpha \in \pi$ and a multiset $\pi_\alpha \subseteq \overline{\pi}$ s.t. $\alpha \xrightarrow{\mathtt{log}} \pi_\alpha$ and $\pi\backslash[\alpha] = \overline{\pi}\backslash\pi_\alpha$[19]. We have already proved that $|\alpha|_{\mathtt{log}} > |\pi_\alpha|_{\mathtt{log}}$, so:

$$\begin{array}{rcl} |\pi|_{\mathtt{log}} & = & |\pi\backslash[\alpha]|_{\mathtt{log}} + |\alpha|_{\mathtt{log}} \\ & = & |\overline{\pi}\backslash\pi_\alpha|_{\mathtt{log}} + |\alpha|_{\mathtt{log}} \\ & > & |\overline{\pi}\backslash\pi_\alpha|_{\mathtt{log}} + |\pi_\alpha|_{\mathtt{log}} \\ & = & |\overline{\pi}|_{\mathtt{log}} \end{array}$$

□

Notice that the proof above uses the property that after a $\xrightarrow{\mathtt{log}}$ step the number of elements of the multiset of sgth of an sps cannot increase: if $\pi \xrightarrow{\mathtt{log}} \overline{\pi}$, then $\mathrm{sgth}(\overline{\pi})$ has at most the same cardinality as $\mathrm{sgth}(\pi)$. It is remarkable that this property fails under $\xrightarrow{\mathtt{str}}$: for example, if $\pi \xrightarrow{\mathtt{str}} \overline{\pi}$ is obtained by a $(!/?c)$ step then $\mathrm{sgth}(\overline{\pi})$ might have more elements than $\mathrm{sgth}(\pi)$, since we have duplicated the possible choices one has to do on the duplicated !-link in order to obtain $\mathrm{sgth}(\overline{\pi})$. The increase of the number of sgth in $\overline{\pi}$ entails the increase of the $\mathtt{log}$-measure: in the $(!/?c)$ case, we can very well have $|\pi|_{\mathtt{log}} < |\overline{\pi}|_{\mathtt{log}}$.

3.2. The strong normalization of $\xrightarrow{\mathtt{str}}$

We now want to prove that $\xrightarrow{\mathtt{str}}$ enjoys SN on sps satisfying AC (Prop. 3.10). As already mentioned in the Introduction, the delicate point behind Prop. 3.10 is that the reduction of a cut t may affect the reduction of other cuts, even at the same exponential depth as t. The critical pairs presented in the proof of Lemma 4.14 of Section 4 are examples of this "meddling" of a $\mathtt{str}$-reduction step in another $\mathtt{str}$-reduction step, the most evident example being the slice α of case 2 of the mentioned proof: in that case the reduction of the cut t even changes the type of the cut r from $(!/!)$ to $(!/?c)$.

[18] This case motivates our variant Definition 3.2 of sgth: if exactly one slice were associated with every !-link, in this case we would have $\mathrm{sgth}(\pi) = \mathrm{sgth}(\overline{\pi}) = \emptyset$ and nothing would decrease.

[19] We use here the standard set notation for multisets of slices: for example $\pi\backslash[\alpha]$ is the multiset π of slices without one occurrence of the slice α.

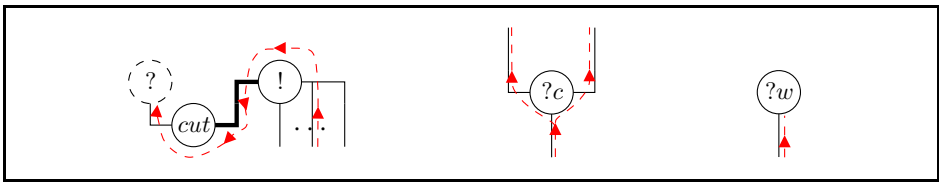


Figure 14: Behaviour of an exponential path meeting structural links (i.e. !-, $?w$-, $?c$-links) and cuts (it stops on all the other links); on $?c$-links a path can choose which premise to follow

The key ingredient we found to tame this meddling of cuts in the life of their fellows with the same exponential depth is the following notion of *exponential dependence* (Definition 3.5).

- given two cuts at the same exponential depth t and r, the `str`-reduction of t affects the `str`-reduction of r only if the premises of r "depend exponentially" on the premises of t;
- exponential dependence is (in some sense) stable under `str`-reduction.

Definition 3.5 (Exponential dependence). Let π be an sps, and let ϕ be an oriented path in a flat of π. We say that ϕ is *exponential* when (see Figure 14):

- ϕ crosses only structural links (i.e. links of type !, $?w$ or $?c$) and cuts;
- if ϕ crosses an edge c upwards, then c is an edge (conclusion or premise) of a structural ?-link (i.e. a link of type $?w$ or $?c$) or an auxiliary conclusion of a !-link;
- if ϕ crosses an edge c downwards, then c is the main conclusion of a !-link which is also premise of a cut link.

Given two edges a and b, we say that a *exponentially depends on* b, whenever there is an exponential path from a to b. Given an edge a, we denote by $\mathrm{pred}(a)$ the set of the *immediate predecessors of* a, i.e. the set of those edges b such that there is an exponential path from a to b crossing exactly one node.

The following fact allows to make induction on the maximal length of the exponential paths starting from a given edge[20]:

Fact 3.6. *If π satisfies* AC*, then every exponential path of π is finite.*

[20] One could also make induction on the following well-founded partial order on the edges of an sps satisying AC: $a \geq_{exp} b$ whenever there is an exponential path from a to b. However, the proof that $\geq_{exp}$ is indeed a partial order is not immediate, and this paper does not lack delicate proofs...

PROOF. Immediate consequence of the fact that every exponential path of π is a path of some switching of π (remember Definition 2.15): hence if π satisfies AC, then π has no exponential cycle. □

Given an sps π satisfying AC, we define two functions (wd^π and $\ln^\pi$) associating an integer with every edge a (resp. !-link o) of π. What happens (as the reader will see during the proof of Proposition 3.10), is that for every !-link o at depth 0 of π, on the one hand $\mathrm{wd}^\pi(o)$ is an upper bound for the number of times o can be copied during a `str`-reduction sequence starting from π, and on the other hand $\ln^\pi(o)$ is an upper bound for the depth of o's residues during a `str`-reduction sequence starting from π. A first evidence that the length of the `str`-reduction sequences starting from an sps π is bounded, is the existence of these two functions: the function wd^π "measures" the maximal "width" of π's reducts while the function $\ln^\pi$ "measures" the maximal depth of π's reducts.

Definition 3.7. Let π be an sps satisfying AC and let a be an edge of π, at any depth. We define $\mathrm{wd}^\pi(a)$, the *width* of a in π, and $\ln^\pi(a)$, the *length* of a in π, by double induction: the first parameter is $\mathrm{depth}(\pi)$, the second one is the maximal length of the exponential paths of π starting from a.

$$\ln^\pi(a) := \begin{cases} 1 + \sup\limits_{\gamma\in\rho}\big(\ln^\gamma(a^\gamma)\big) + \ln^\pi(b) & \text{if } a \text{ is an auxiliary conclusion of a !-link with main conclusion } b\text{, box } \rho \text{ and } a^\gamma \text{ is the conclusion corresponding to } a \text{ of the slice } \gamma\in\rho, \\ 1 + \sup\limits_{b\in\mathrm{pred}(a)}\big(\ln^\pi(b)\big) & \text{otherwise.} \end{cases}$$

$$\mathrm{wd}^\pi(a) := \begin{cases} 1 & \text{if } \mathrm{pred}(a)=\emptyset, \\ \big(1+\sum\limits_{\gamma\in\rho}\mathrm{wd}^\gamma(a^\gamma)\big)\,\mathrm{wd}^\pi(b) & \text{if } a \text{ is an auxiliary conclusion of a !-link with main conclusion } b\text{, box } \rho \text{ and } a^\gamma \text{ is the conclusion corresponding to } a \text{ of the slice } \gamma\in\rho, \\ \sum\limits_{b\in\mathrm{pred}(a)}\mathrm{wd}^\pi(b) & \text{otherwise.} \end{cases}$$

We extend wd^π and $\ln^\pi$ to the !-links of π, by setting for a !-link o with main conclusion b: $\ln^\pi(o)=\ln^\pi(b)$ and $\mathrm{wd}^\pi(o)=\mathrm{wd}^\pi(b)$.

We will often simply write $\ln(a)$ or $\mathrm{wd}(a)$ (resp. $\ln(o)$ or $\mathrm{wd}(o)$), when there is no ambiguity on the sps π we refer to.

If a is an auxiliary conclusion of a !-link o of an sps satisfying AC, then o's main conclusion is the unique immediate predecessor of a, so that $\mathrm{pred}(a)\neq\emptyset$: this is used in the previous definition of wd. Notice also that by definition $\ln(a)>0$ and $\mathrm{wd}(a)>0$. Finally the long-awaited definition of `str`-measure:

Definition 3.8 (`str`-measure). The `str`-measure of an sps π satisfying AC is a natural number, denoted by $|\pi|_{\mathtt{str}}$ and defined by induction on $\mathrm{depth}(\pi)$, as follows:

$$|\pi|_{\mathtt{str}} \quad := \quad \sum_{o \in !^0(\pi)} \left(\mathrm{wd}^{\pi}(o)(\ln^{\pi}(o) + |\pi^o|_{\mathtt{str}}) \right)$$

where $!^0(\pi)$ denotes the set of !-links at depth 0 of π, and π^0 denotes the sps associated with the !-link o (see Definition 2.2).

Consider for example the slice δ of Fig. 9, we have $\ln^{\delta}(b) = 1$, $\ln^{\delta}(a) = 1+1+1 = 3$ and $\ln^{\delta}(d) = 1 + \sup\{1,3\} = 4$; as for the width, $\mathrm{wd}^{\delta}(b) = 1$, $\mathrm{wd}^{\delta}(a) = 2$, $\mathrm{wd}^{\delta}(d) = 3$. Thus $|\delta|_{\mathtt{str}} = \mathrm{wd}^{\delta}(b) \cdot \left(\ln^{\delta}(b) + 0\right) = 1$. Then consider the slice $\delta\delta$ of Fig. 9, we have $\ln^{\delta\delta}(e) = 1 + \ln^{\delta\delta}(d) = 1 + \ln^{\delta}(d) = 5$ and $\mathrm{wd}^{\delta\delta}(e) = \mathrm{wd}^{\delta\delta}(d) = \mathrm{wd}^{\delta}(d) = 3$. So that $|\delta\delta|_{\mathtt{str}} = 1 + \mathrm{wd}^{\delta\delta}(e)\left(\ln^{\delta\delta}(e) + |\delta|_{\mathtt{str}}\right) = 1 + 3(5+1) = 19$. The reader can check that the `str`-reducts of $\delta\delta$ (i.e. the slices pictured in the above line of Fig. 10) have a `str`-measure stricly less than $|\delta\delta|_{\mathtt{str}}$. Moreover, notice that the reduction $\delta\delta \xrightarrow{\mathtt{cut+}} \delta\delta$ contains *log*-steps (the 3 ones represented in the lower part of Fig. 10).

The following fact shows some kind of "modularity" and of "monotonicity" of the functions $\ln$ and wd, two ingredients of Proposition 3.10. Recall the notion of module of Definition 2.11.

Fact 3.9 (Modularity). *Let $\alpha = \omega[\beta]$ and $\alpha' = \omega[\beta']$ be two switching acyclic slices, where the module β' replaces the module β in α'. For every pending edge (i.e. conclusion or hypothesis) b of β, denote by b' the corresponding pending edge of β'. The following properties hold:*

1. *let b be a pending edge of β and suppose that b exponentially depends in $\omega[\beta]$ only on edges of $\omega[\,]$, i.e. if b depends on $c \neq b$, then c is not an edge of β. Then $\ln^{\alpha}(b) = \ln^{\alpha'}(b')$ and $\mathrm{wd}^{\alpha}(b) = \mathrm{wd}^{\alpha'}(b')$;*
2. *if d is an edge of α which is not an edge of β and if for every pending edge b of β we have $\ln^{\alpha}(b) = \ln^{\alpha'}(b')$ (resp. $\ln^{\alpha}(b) \geq \ln^{\alpha'}(b')$), then $\ln^{\alpha}(d) = \ln^{\alpha'}(d)$ (resp. $\ln^{\alpha}(d) \geq \ln^{\alpha'}(d)$); the same holds for wd.*

PROOF. It is a consequence of Definition 3.7. □

As for the `log`-measure, the `str`-measure of π is the sum of the `str`-measures of the slices of π, i.e. $|\pi|_{\mathtt{str}} = \sum_{\alpha \in \pi} |\alpha|_{\mathtt{str}}$.

Proposition 3.10. *The reduction $\xrightarrow{\mathtt{str}}$ is* SN *on the sps satisfying* AC.

PROOF. For π satisfying AC, we prove that $\pi \xrightarrow{\mathtt{str}} \overline{\pi}$ implies $|\pi|_{\mathtt{str}} > |\overline{\pi}|_{\mathtt{str}}$. We restrict to the case π has exactly one slice α: the extension to the general case is obtained exactly as in the proof of Proposition 3.4.

So let α be a switching acyclic slice and $\alpha \xrightarrow{\mathtt{str}} \pi_\alpha$. First remark that π_α contains a unique slice (a glance at Figure 5 will convince the reader). Let then $\pi_\alpha = \{\alpha'\}$ and let t be the cut link of α reduced during the $\xrightarrow{\mathtt{str}}$ step under consideration. Our goal is to show $|\alpha|_{\mathtt{str}} > |\alpha'|_{\mathtt{str}}$. Actually we prove a stronger statement, by induction on the depth of t:

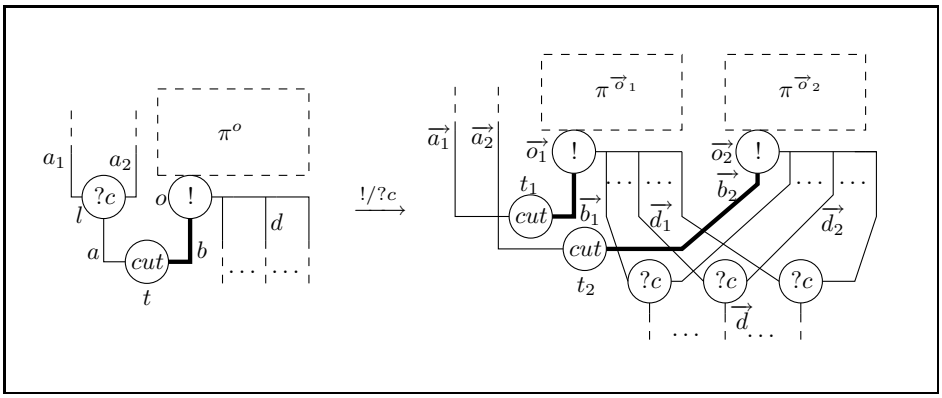


Figure 15: The $(!/?c)$ case of the proof of Proposition 3.10

(1) $|\alpha|_{\mathtt{str}} > |\alpha'|_{\mathtt{str}}$,

(2) for every conclusion d of α, let $\overrightarrow{d}$ be the conclusion of α' corresponding to d, one has $\ln^{\alpha}(d) \geq \ln^{\alpha'}(\overrightarrow{d})$ and $\mathrm{wd}^{\alpha}(d) \geq \mathrm{wd}^{\alpha'}(\overrightarrow{d})$.

Base of induction. If t has depth 0, then we have three cases, depending on the type of t.

Case $(!/?c)$**.** If t is a cut of type $(!/?c)$, then (see Figure 15) let l (resp. o) be the $?c$-link (resp. !-link) whose conclusion a (resp. whose main conclusion b) is a premise of t, let a_1, a_2 be the premises of l, and let $\overrightarrow{o_1}, \overrightarrow{o_2}$ be the two copies of o in α': $\overrightarrow{b_i}$ and $\overrightarrow{a_i}$, $i \in \{1, 2\}$, are premises of a cut link t_i.

Let S be the set of !-links at depth 0 of α different from o, i.e. $!^0(\alpha) = \{o\} \cup S$ (and the union is actually a disjoint union). Observe that every !-link $v \in S$ has a unique residue $\overrightarrow{v}$ in α': we have $!^0(\alpha') = \{\overrightarrow{o_1}, \overrightarrow{o_2}\} \cup \overrightarrow{S}$. We use in the sequel the notation π^v for the box associated with the !-link v (see Definition 2.2). Let us define:

$$|S|_{\mathtt{str}} := \sum_{v \in S} \big(\mathrm{wd}^{\alpha}(v)(\ln^{\alpha}(v) + |\pi^v|_{\mathtt{str}})\big)$$
$$|\overrightarrow{S}|_{\mathtt{str}} := \sum_{\overrightarrow{v} \in \overrightarrow{S}} \big(\mathrm{wd}^{\alpha'}(\overrightarrow{v})(\ln^{\alpha'}(\overrightarrow{v}) + |\pi^{\overrightarrow{v}}|_{\mathtt{str}})\big) \quad .$$

Then by definition we have:

$$|\alpha|_{\mathtt{str}} = \mathrm{wd}^{\alpha}(o)(\ln^{\alpha}(o) + |\pi^o|_{\mathtt{str}}) + |S|_{\mathtt{str}}$$
$$|\alpha'|_{\mathtt{str}} = \mathrm{wd}^{\alpha'}(\overrightarrow{o_1})\big(\ln^{\alpha'}(\overrightarrow{o_1}) + \pi^{\overrightarrow{o_1}}\big) + \mathrm{wd}^{\alpha'}(\overrightarrow{o_2})\big(\ln^{\alpha'}(\overrightarrow{o_2}) + \pi^{\overrightarrow{o_2}}\big) + |\overrightarrow{S}|_{\mathtt{str}}.$$

Consider any conclusion d of o in α different from b, denote as $\overrightarrow{d}$ the conclusion of the $?c$-link created in α' and corresponding to d, and denote as $\overrightarrow{d_1}, \overrightarrow{d_2}$ the two premises of this $?c$-link, one being an auxiliary conclusion of $\overrightarrow{o_1}$, the other one being an auxiliary conclusion of $\overrightarrow{o_2}$ (see Figure 15). We prove $\ln^{\alpha}(d) = \ln^{\alpha'}(\overrightarrow{d})$ and $\mathrm{wd}^{\alpha}(d) = \mathrm{wd}^{\alpha'}(\overrightarrow{d})$.

Since α satisfies AC, the edge a_i, $i \in \{1,2\}$, does not exponentially depend on any edge involved in the reduction of t, so by Fact 3.9-1, one has $\ln(\overrightarrow{a_i}) = \ln(a_i)$ and $\mathrm{wd}(\overrightarrow{a_i}) = \mathrm{wd}(a_i)$. Then the following equalities hold, where for $\gamma \in \pi^o$ (resp. $\gamma \in \pi^{\overrightarrow{o_i}}$) we denote by d^γ the conclusion of γ corresponding to the auxiliary conclusion d of o (resp. of $\overrightarrow{o_i}$):

$$\begin{array}{rcl}
\ln^\alpha(d) & = & 1 + \sup_{\gamma\in\pi^o}\big(\ln^\gamma(d^\gamma)\big) + \ln^\alpha(b) \\
& = & 2 + \sup_{\gamma\in\pi^o}\big(\ln^\gamma(d^\gamma)\big) + \ln^\alpha(a) \\
& = & 3 + \sup_{\gamma\in\pi^o}\big(\ln^\gamma(d^\gamma)\big) + \sup\{\ln(a_1),\ln(a_2)\} \\
& = & 1 + \sup\left\{ \begin{array}{l} \big(\sup_{\gamma\in\pi^{\overrightarrow{o_1}}}\big(\ln^\gamma(d^\gamma)\big) + \ln(\overrightarrow{a_1}) + 2\big), \\ \big(\sup_{\gamma\in\pi^{\overrightarrow{o_2}}}\big(\ln^\gamma(d^\gamma)\big) + \ln(\overrightarrow{a_2}) + 2\big) \end{array} \right\} \\
& = & 1 + \sup\{\ln^{\alpha'}(\overrightarrow{d_1}), \ln^{\alpha'}(\overrightarrow{d_2})\} \\
& = & \ln^{\alpha'}(\overrightarrow{d}).
\end{array}$$

As for $\mathrm{wd}^\alpha(d) = \mathrm{wd}^{\alpha'}(\overrightarrow{d})$:

$$\begin{array}{rcl}
\mathrm{wd}^\alpha(d) & = & \big(1 + \sum_{\gamma\in\pi^o} \mathrm{wd}^\gamma(d^\gamma)\big)\,\mathrm{wd}^\alpha(b) \\
& = & \big(1 + \sum_{\gamma\in\pi^o} \mathrm{wd}^\gamma(d^\gamma)\big)\,\mathrm{wd}^\alpha(a) \\
& = & \big(1 + \sum_{\gamma\in\pi^o} \mathrm{wd}^\gamma(d^\gamma)\big)\big(\mathrm{wd}(a_1) + \mathrm{wd}(a_2)\big) \\
& = & \big(1 + \sum_{\gamma\in\pi^{\overrightarrow{o_1}}} \mathrm{wd}^\gamma(d^\gamma)\big)\,\mathrm{wd}(\overrightarrow{a_1}) + \big(1 + \sum_{\gamma\in\pi^{\overrightarrow{o_2}}} \mathrm{wd}^\gamma(d^\gamma)\big)\,\mathrm{wd}(\overrightarrow{a_2}) \\
& = & \big(1 + \sum_{\gamma\in\pi^{\overrightarrow{o_1}}} \mathrm{wd}^\gamma(d^\gamma)\big)\,\mathrm{wd}(\overrightarrow{b_1}) + \big(1 + \sum_{\gamma\in\pi^{\overrightarrow{o_2}}} \mathrm{wd}^\gamma(d^\gamma)\big)\,\mathrm{wd}(\overrightarrow{b_2}) \\
& = & \mathrm{wd}(\overrightarrow{d_1}) + \mathrm{wd}(\overrightarrow{d_2}) \\
& = & \mathrm{wd}^{\alpha'}(\overrightarrow{d}).
\end{array}$$

This implies by Fact 3.9-2, that for every $v \in S$ one has $\ln(v) = \ln(\overrightarrow{v})$ and $\mathrm{wd}(v) = \mathrm{wd}(\overrightarrow{v})$, and that for every conclusion d of α, $\ln^\alpha(d) = \ln^{\alpha'}(\overrightarrow{d})$ and $\mathrm{wd}^\alpha(d) = \mathrm{wd}^{\alpha'}(\overrightarrow{d})$: in particular (2) holds.

As for (1), notice that $\ln(o) = 1 + \sup\{\ln(\overrightarrow{o_1}), \ln(\overrightarrow{o_2})\}$, so that $\ln(o) > \ln(\overrightarrow{o_i})$ ($i \in \{1,2\}$). Notice also that $\mathrm{wd}(o) = \mathrm{wd}(\overrightarrow{o_1}) + \mathrm{wd}(\overrightarrow{o_2})$, as well as $\pi^o = \pi^{\overrightarrow{o_i}}$ ($i \in \{1,2\}$). We can then deduce[21]

$$\begin{array}{rcl}
\mathrm{wd}(o)(\ln(o) + |\pi^o|_{\mathtt{str}}) & > & \mathrm{wd}(\overrightarrow{o_1})(\ln(\overrightarrow{o_1}) + |\pi^{\overrightarrow{o_1}}|_{\mathtt{str}}) \\
& & + \mathrm{wd}(\overrightarrow{o_2})(\ln(\overrightarrow{o_2}) + |\pi^{\overrightarrow{o_2}}|_{\mathtt{str}}).
\end{array}$$

On the other hand, for every $v \in S$ the box associated with v in π is the same sps as the box associated with $\overrightarrow{v}$ in α', so that $|\pi^v|_{\mathtt{str}} = |\pi^{\overrightarrow{v}}|_{\mathtt{str}}$. This means that $|S|_{\mathtt{str}} = |\overrightarrow{S}|_{\mathtt{str}}$. This equality and the inequality above yield $|\alpha|_{\mathtt{str}} > |\alpha'|_{\mathtt{str}}$.

Case (!/!)**.** If t is of type (!/!), then (see Figure 16) let o (resp. u) be the !-link of which the main (resp. an auxiliary) conclusion a (resp. b) is a premise of t, let c be the

[21] We use here the fact that for every edge a of π, one has $\mathrm{wd}(a) > 0$, as already mentioned after Definition 3.7.

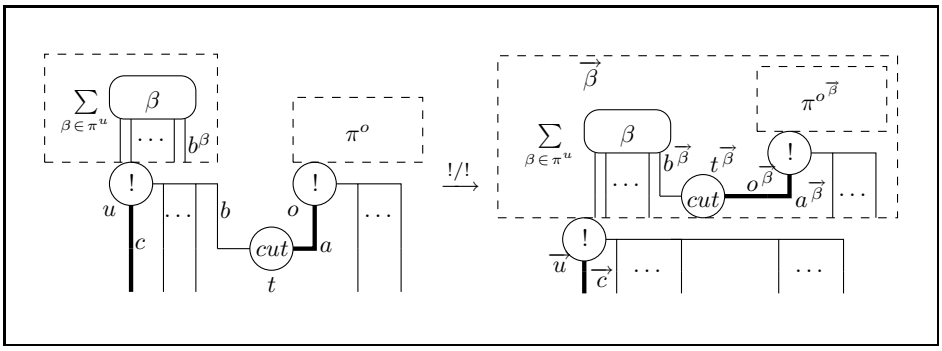


Figure 16: The $(!/!)$ case of the proof of Proposition 3.10

main conclusion of u, let $\overrightarrow{u}$ be the residue of u in α' and for every slice $\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}$ (if any) let $o^{\overrightarrow{\beta}}$ be the copy of o which "entered" the !-link u during the reduction of t and finally let $t^{\overrightarrow{\beta}}$ be the cut created in $\overrightarrow{\beta}$ by the reduction of t.

We set $!^0(\alpha) = \{u, o\} \cup S$, i.e. S is the set of !-links at depth 0 of α different from u and o. Observe that each !-link $v \in S$ has a unique residue $\overrightarrow{v}$ in α', so that $!^0(\alpha') = \{\overrightarrow{u}\} \cup \overrightarrow{S}$. Let us define:

$$\begin{array}{lcl} |S|_{\mathtt{str}} & := & \sum_{v\in S}\big(\mathrm{wd}^\alpha(v)(\mathrm{ln}^\alpha(v) + |\pi^v|_{\mathtt{str}})\big) \\ |\overrightarrow{S}|_{\mathtt{str}} & := & \sum_{\overrightarrow{v}\in\overrightarrow{S}}\big(\mathrm{wd}^{\alpha'}(\overrightarrow{v})(\mathrm{ln}^{\alpha'}(\overrightarrow{v}) + |\pi^{\overrightarrow{v}}|_{\mathtt{str}})\big). \end{array}$$

Then by definition we have:

$$\begin{array}{lcl} |\alpha|_{\mathtt{str}} & = & \mathrm{wd}(u)\big(\mathrm{ln}(u) + |\pi^u|_{\mathtt{str}}\big) + \mathrm{wd}(o)\big(\mathrm{ln}(o) + |\pi^o|_{\mathtt{str}}\big) + |S|_{\mathtt{str}} \\ |\alpha'|_{\mathtt{str}} & = & \mathrm{wd}(\overrightarrow{u})\big(\mathrm{ln}(\overrightarrow{u}) + |\pi^{\overrightarrow{u}}|_{\mathtt{str}}\big) + |\overrightarrow{S}|_{\mathtt{str}}. \end{array}$$

Consider any conclusion d of u in α different from b, and denote as $\overrightarrow{d}$ the conclusion of $\overrightarrow{u}$ in α' corresponding to d. If $d = c$, then d does not exponentially depend on any edge involved in the reduction of t: so by Fact 3.9-1, one has $\mathrm{ln}^\alpha(d) = \mathrm{ln}^{\alpha'}(\overrightarrow{d})$ and $\mathrm{wd}^\alpha(d) = \mathrm{wd}^{\alpha'}(\overrightarrow{d})$. If $d \neq c$, let π^u (resp. $\pi^{\overrightarrow{u}}$) be the box of u (resp. $\overrightarrow{u}$). By Definition 3.7 (with the same notations), one has:

$$\begin{array}{lcl} \mathrm{ln}^\alpha(d) & := & 1 + \sup_{\gamma\in\pi^u}\big(\mathrm{ln}^\gamma(a^\gamma)\big) + \mathrm{ln}^\alpha(c) \\ \mathrm{ln}^{\alpha'}(\overrightarrow{d}) & := & 1 + \sup_{\overrightarrow{\gamma}\in\pi^{\overrightarrow{u}}}\big(\mathrm{ln}^{\overrightarrow{\gamma}}(a^{\overrightarrow{\gamma}})\big) + \mathrm{ln}^{\alpha'}(\overrightarrow{c}). \end{array}$$

Notice that every edge g conclusion of π^u exponentially depends only on edges which are not involved in the reduction of t, so by Fact 3.9-1 we have $\mathrm{ln}^{\pi^u}(g) = \mathrm{ln}^{\pi^{\overrightarrow{u}}}(\overrightarrow{g})$.[22] Since we already noticed that $\mathrm{ln}^\alpha(c) = \mathrm{ln}^\alpha(\overrightarrow{c})$, we can eventually conclude that for

[22] Notice that we are actually using a slight variant of Fact 3.9-1: with the notations of Fact 3.9, some pending edges of β and β' do not coincide. These edges are conclusions of α and α' and it is easy to check that Fact 3.9-1 holds in this case too.

every conclusion d of u in π different from b one has $\ln(d) = \ln(\overrightarrow{d})$, and in the same way one obtains the equality $\mathrm{wd}(d) = \mathrm{wd}(\overrightarrow{d})$. In particular, $\ln(u) = \ln(\overrightarrow{u})$ and $\mathrm{wd}(u) = \mathrm{wd}(\overrightarrow{u})$.

Then, consider any auxiliary conclusion d of o in α, and denote by $\overrightarrow{d}$ the conclusion of $\overrightarrow{u}$ in α' corresponding to d. Also in this case we will prove $\ln^{\alpha}(d) = \ln^{\alpha'}(\overrightarrow{d})$ and $\mathrm{wd}^{\alpha}(d) \geq \mathrm{wd}^{\alpha'}(\overrightarrow{d})$. Still by Fact 3.9-1 we have that $\ln^{\beta}(b^{\beta}) = \ln^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})$ and $\mathrm{wd}^{\beta}(b^{\beta}) = \mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})$.[23] The following equalities hold (where, like in the $(!/?c)$-case, d^{γ} denotes the conclusion of the slice γ corresponding to d):

$$\begin{array}{rcl}
\ln^{\alpha}(d) & = & 1 + \sup_{\gamma \in \pi^{o}} \left(\ln^{\gamma}(d^{\gamma})\right) + \ln^{\alpha}(a) \\
& = & 2 + \sup_{\gamma \in \pi^{o}} \left(\ln^{\gamma}(d^{\gamma})\right) + \ln^{\alpha}(b) \\
& = & 3 + \sup_{\gamma \in \pi^{o}} \left(\ln^{\gamma}(d^{\gamma})\right) + \sup_{\beta \in \pi^{u}} \left(\ln^{\beta}(b^{\beta})\right) + \ln^{\alpha}(c) \\
& = & 3 + \sup_{\gamma \in \pi^{o}} \left(\ln^{\gamma}(d^{\gamma})\right) + \sup_{\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}} \left(\ln^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})\right) + \ln^{\alpha'}(\overrightarrow{c}) \\
& = & 3 + \sup_{\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}} \left(\sup_{\gamma \in \pi^{o^{\overrightarrow{\beta}}}} \left(\ln^{\gamma}(d^{\gamma})\right) + \ln^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})\right) + \ln^{\alpha'}(\overrightarrow{c}) \\
& = & 2 + \sup_{\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}} \left(\sup_{\gamma \in \pi^{o^{\overrightarrow{\beta}}}} \left(\ln^{\gamma}(d^{\gamma})\right) + \ln^{\overrightarrow{\beta}}(a^{\overrightarrow{\beta}})\right) + \ln^{\alpha'}(\overrightarrow{c}) \\
& = & 1 + \sup_{\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}} \left(\ln^{\overrightarrow{\beta}}(d^{\overrightarrow{\beta}})\right) + \ln^{\alpha'}(\overrightarrow{c}) \\
& = & \ln^{\alpha'}(\overrightarrow{d}).
\end{array}$$

In a similar way we prove that $\mathrm{wd}^{\alpha}(d) \geq \mathrm{wd}^{\alpha'}(\overrightarrow{d})$:

[23] See footnote 22.

$$
\begin{aligned}
\mathrm{wd}^{\alpha}(d) &= \Big(1+\sum_{\gamma\in\pi^o}\mathrm{wd}^{\gamma}(d^{\gamma})\Big)\,\mathrm{wd}^{\alpha}(a)\\
&= \Big(1+\sum_{\gamma\in\pi^o}\mathrm{wd}^{\gamma}(d^{\gamma})\Big)\,\mathrm{wd}^{\alpha}(b)\\
&= \Big(1+\sum_{\gamma\in\pi^o}\mathrm{wd}^{\gamma}(d^{\gamma})\Big)\Big(1+\sum_{\beta\in\pi^u}\mathrm{wd}^{\beta}(b^{\beta})\Big)\,\mathrm{wd}^{\alpha}(c)\\
&= \Big(1+\sum_{\gamma\in\pi^o}\mathrm{wd}^{\gamma}(d^{\gamma})\Big)\Big(1+\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})\Big)\,\mathrm{wd}^{\alpha'}(\overrightarrow{c})\\
&= \left(1+\sum_{\gamma\in\pi^o}\mathrm{wd}^{\gamma}(d^{\gamma})+\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})+\sum_{\substack{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}\\ \gamma\in\pi^o}}\mathrm{wd}^{\gamma}(d^{\gamma})\,\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})\right)\mathrm{wd}^{\alpha'}(\overrightarrow{c})\\
&\geq \left(1+\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})+\sum_{\substack{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}\\ \gamma\in\pi^{o\overrightarrow{\beta}}}}\mathrm{wd}^{\gamma}(d^{\gamma})\,\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})\right)\mathrm{wd}^{\alpha'}(\overrightarrow{c})\\
&= \left(1+\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\Big(1+\sum_{\gamma\in\pi^{o\overrightarrow{\beta}}}\mathrm{wd}^{\gamma}(d^{\gamma})\Big)\,\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})\right)\mathrm{wd}^{\alpha'}(\overrightarrow{c})\\
&= \Big(1+\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(d^{\overrightarrow{\beta}})\Big)\,\mathrm{wd}^{\alpha'}(\overrightarrow{c})\\
&= \mathrm{wd}^{\alpha'}(\overrightarrow{d}).
\end{aligned}
$$

Like in the $(!/?c)$-case, all this implies by Fact 3.9-2, that for every $v\in S$ one has $\ln(v)=\ln(\overrightarrow{v})$ and $\mathrm{wd}(v)\geq\mathrm{wd}(\overrightarrow{v})$, and that for every conclusion d of α, $\ln^{\alpha}(d)=\ln^{\alpha'}(\overrightarrow{d})$ and $\mathrm{wd}^{\alpha}(d)\geq\mathrm{wd}^{\alpha'}(\overrightarrow{d})$: in particular (2) holds.

As for (1), we first prove that

$$
\mathrm{wd}(\overrightarrow{u})\big(\ln(\overrightarrow{u})+|\pi^{\overrightarrow{u}}|_{\mathtt{str}}\big)<\mathrm{wd}(u)\big(\ln(u)+|\pi^{u}|_{\mathtt{str}}\big)+\mathrm{wd}(o)\big(\ln(o)+|\pi^{o}|_{\mathtt{str}}\big)\ .
$$

In case π^u is the empty set of slices, the inequality holds: $|\pi^{\overrightarrow{u}}|_{\mathtt{str}}=|\pi^{u}|_{\mathtt{str}}=0$, so that $\mathrm{wd}(\overrightarrow{u})\big(\ln(\overrightarrow{u})+|\pi^{\overrightarrow{u}}|_{\mathtt{str}}\big)=\mathrm{wd}(u)\big(\ln(u)+|\pi^{u}|_{\mathtt{str}}\big)<\mathrm{wd}(u)\big(\ln(u)+|\pi^{u}|_{\mathtt{str}}\big)+\mathrm{wd}(o)\big(\ln(o)+|\pi^{o}|_{\mathtt{str}}\big)$ (since $\mathrm{wd}(o),\ln(o),|\pi^{o}|_{\mathtt{str}}>0$). In case π^u is not empty, for $\beta\in\pi^u$ (resp. $\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}$), define $H_\beta=2+\ln^{\beta}(b^{\beta})+|\pi^{o}|_{\mathtt{str}}$ (resp. $H_{\overrightarrow{\beta}}=2+\ln^{\beta}(b^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}$): notice that $H_\beta=H_{\overrightarrow{\beta}}>0$ (we are using $\pi^u\neq 0$), and this will be used in the following sequence of equalities and inequalities. By definition $\ln(o)=2+\ln(u)+\sup_{\beta\in\pi^u}\big(\ln^{\beta}(b^{\beta})\big)$, so we have (using Fact 3.9):

$$
\begin{aligned}
\mathrm{wd}(o)(\ln(o)+|\pi^o|_{\mathtt{str}}) &\geq \mathrm{wd}(o)\big(2+\sup_{\beta\in\pi^u}\big(\ln^\beta(b^\beta)\big)+|\pi^o|_{\mathtt{str}}\big)\\
&= \mathrm{wd}(u)\Big(1+\sum_{\beta\in\pi^u}\mathrm{wd}^\beta(b^\beta)\Big)\Big(\sup_{\beta\in\pi^u}H_\beta\Big)\\
&\geq \mathrm{wd}(u)\Big(\sup_{\beta\in\pi^u}H_\beta\Big)+\mathrm{wd}(u)\Big(\sum_{\beta\in\pi^u}\mathrm{wd}^\beta(b^\beta)H_\beta\Big)\\
&= \mathrm{wd}(\overrightarrow{u})\Big(\sup_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}H_{\overrightarrow{\beta}}\Big)+\mathrm{wd}(\overrightarrow{u})\Big(\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})H_{\overrightarrow{\beta}}\Big)\\
&= \mathrm{wd}(\overrightarrow{u})\Big(\sup_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}H_{\overrightarrow{\beta}}\Big)+\mathrm{wd}(\overrightarrow{u})\Big(\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})H_{\overrightarrow{\beta}}\Big)\\
&> \mathrm{wd}(\overrightarrow{u})\Big(\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})H_{\overrightarrow{\beta}}\Big)\\
&= \mathrm{wd}(\overrightarrow{u})\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\Big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(2+\ln^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\Big)\\
&= \big(\mathrm{wd}(\overrightarrow{u})\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big)\\
&\qquad+\mathrm{wd}(\overrightarrow{u})\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\Big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(1+\ln^{\overrightarrow{\beta}}(b^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\Big)\\
&= \big(\mathrm{wd}(\overrightarrow{u})\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big)\\
&\qquad+\mathrm{wd}(\overrightarrow{u})\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\Big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(\ln^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\Big)\\
&> \mathrm{wd}(\overrightarrow{u})\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(\ln^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\big).
\end{aligned}
$$

Moreover, since $|\pi^u|_{\mathtt{str}}=\sum_{\beta\in\pi^u}|\beta|_{\mathtt{str}}$, we have:

$$
\begin{aligned}
|\pi^{\overrightarrow{u}}|_{\mathtt{str}} &= \textstyle\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\big(|\beta|_{\mathtt{str}}+\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(\ln^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\big)\\
&= |\pi^u|_{\mathtt{str}}+\textstyle\sum_{\overrightarrow{\beta}\in\pi^{\overrightarrow{u}}}\big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(\ln^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})+|\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\big).
\end{aligned}
$$

Now remember that $\mathrm{wd}(u)=\mathrm{wd}(\overrightarrow{u})$ and $\ln(u)=\ln(\overrightarrow{u})$. Thus:

$$
\begin{aligned}
\mathrm{wd}(\overrightarrow{u})(\ln(\overrightarrow{u}) + |\pi^{\overrightarrow{u}}|_{\mathtt{str}}) &= \mathrm{wd}(u)(\ln(u) + |\pi^{\overrightarrow{u}}|_{\mathtt{str}}) \\
&= \mathrm{wd}(u)\Bigg(\ln(u) + |\pi^{u}|_{\mathtt{str}} \\
&\qquad + \sum_{\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}} \Big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})\big(\ln^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}}) + |\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}}\big)\Big)\Bigg) \\
&= \mathrm{wd}(u)\big(\ln(u) + |\pi^{u}|_{\mathtt{str}}\big) \\
&\qquad + \mathrm{wd}(\overrightarrow{u}) \sum_{\overrightarrow{\beta} \in \pi^{\overrightarrow{u}}} \big(\mathrm{wd}^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}})(\ln^{\overrightarrow{\beta}}(o^{\overrightarrow{\beta}}) + |\pi^{o^{\overrightarrow{\beta}}}|_{\mathtt{str}})\big) \\
&< \mathrm{wd}(u)\big(\ln(u) + |\pi^{u}|_{\mathtt{str}}\big) + \mathrm{wd}(o)\big(\ln(o) + |\pi^{o}|_{\mathtt{str}}\big).
\end{aligned}
$$

On the other hand, for every $v \in S$ the box associated with v in α is the same sps as the box associated with $\overrightarrow{v}$ in α', so that $|S|_{\mathtt{str}} = |\overrightarrow{S}|_{\mathtt{str}}$. The previous inequation allows then to conclude $|\alpha|_{\mathtt{str}} > |\alpha'|_{\mathtt{str}}$.

Case $(!/?w)$**.** If t is of type $(!/?w)$, then the case is simple and left to the reader. Notice that in this case the values of length and width can decrease, since the reduction of t erases a !-link.

Inductive step. If t is a cut in a !-link o of α', then α' is obtained by replacing the box π^o associated with o with a box $\overline{\pi^o}$ s.t. $\pi^o \xrightarrow{\mathtt{str}} \overline{\pi^o}$. By induction hypothesis we know that (1) $|\overline{\pi^o}|_{\mathtt{str}} < |\pi^o|_{\mathtt{str}}$, and (2) for every conclusion d of π^o, $\ln^{\pi^o}(d) \geq \ln^{\overline{\pi^o}}(\overrightarrow{d}), \mathrm{wd}^{\pi^o}(d) \geq \mathrm{wd}^{\overline{\pi^o}}(\overrightarrow{d})$.
By Fact 3.9-2, this implies both (1) and (2) for α, α'. □

We conclude the subsection by noting that (in a perfectly symmetric way w.r.t. the `log`-measure) the `str`-measure may increase under $\xrightarrow{\mathtt{log}}$: indeed, the reduction $\xrightarrow{\mathtt{log}}$ can change the exponential paths of an sps and their lengths may increase (think for example of the main conclusion of a !-link o which is the premise of a $?d$-link itself cut with a !-link u: after the reduction of the $(!/?d)$ cut, some exponential paths disappear and some new ones starting from o and crossing edges of u's box might appear and might be longer than the erased ones: this is exactly what happens in the last reduction step of Fig. 10). If the length of exponential paths grows, so do also the values of the functions $\ln$ and wd, and consequently the `str`-measure.

4. Standardization for sliced pure structures

In this section we prove our main result: the standardization theorem for sps (Theorem 4.2). Basically, the standardization theorem reduces the problem of SN to a problem of WN w.r.t. a subreduction of $\xrightarrow{\mathtt{cut}}$, called *non-erasing reduction*. The non-erasing reduction steps have a key property: they never erase cuts different from the reduced one. In fact, the notion of non-erasing reduction has to be defined *very* carefully: not only a non-erasing reduction step does not erase cuts, but also it does not erase (nor change the non-erasing nature of) "future cuts", that is of cuts which can be created during any reduction sequence.[24].

Definition 4.1 (Erasing cut). The *erasing* reduction steps are the following:

- $(\oplus_i/\&_j)$ for $i \neq j$, $(!/?w)$ and $(\top/\mathrm{cc})$,
- $(!/?d)$ (resp. $(!/!)$) in the case the empty sps is associated with the !-link whose main conclusion (resp. auxiliary conclusion) is a premise of the cut to be reduced.

The other reduction steps are called *non-erasing*.

The erasing (resp. non-erasing) reduction, denoted by $\xrightarrow{\mathtt{e}}$ (resp. by $\xrightarrow{\neg\mathtt{e}}$), is the context closure of the union of the erasing (resp. non-erasing) reduction steps. A cut is erasing (resp. non-erasing) when so is its reduction.

Theorem 4.2 (Standardization for sps). *Let π be an sps which satisfies* AC. *If $\pi \in \mathrm{WN}^{\neg\mathtt{e}}$, then $\pi \in \mathrm{SN}$.*

With respect to the description of Gandy's method given in the Introduction, the standardization theorem achieves all the tasks except the proof of $\mathrm{WN}^{\neg\mathtt{e}}$. We split the proof of Theorem 4.2 in two parts: in Subsection 4.1 we prove that SN is a consequence of $\mathrm{SN}^{\neg\mathtt{e}}$ (Proposition 4.5), in Subsection 4.2 we prove the equivalence between $\mathrm{SN}^{\neg\mathtt{e}}$ and $\mathrm{WN}^{\neg\mathtt{e}}$ (Proposition 4.10). This last step is the most delicate one, and uses the key notion of labelled sps and labelled reduction (Definition 4.6 and Definition 4.7).

4.1. SN *is a consequence of* $\mathrm{SN}^{\neg\mathtt{e}}$

In this subsection we prove Proposition 4.5: SN is a consequence of $\mathrm{SN}^{\neg\mathtt{e}}$. The proof is based on two simple facts: (i) erasing steps can always be postponed after non-erasing ones (Lemma 4.4); (ii) there is no infinite sequence of erasing steps (the $\xrightarrow{\mathtt{e}}$ reduction clearly decreases the size of an sps). Then suppose there exists an infinite sequence of $\xrightarrow{\mathtt{cut}}$ steps starting from an sps π (i.e. suppose $\pi \notin \mathrm{SN}$). This sequence should contain an infinite number of non-erasing steps: by iterating Lemma 4.4 we then obtain an arbitrarily long sequence of non-erasing steps starting from π, i.e. $\pi \notin \mathrm{SN}^{\neg\mathtt{e}}$.

Lemma 4.3. *Let π and π' be two sps. If $\pi \xrightarrow{\mathtt{e}} \pi'$, then every cut link t' of π' has an ancestor $\overleftarrow{t'}$ in π. If t' is non erasing, the two cuts t' and $\overleftarrow{t'}$ have the same type.*

[24] This might sound a bit mysterious by now, but it will appear in full light in Subsection 5.2.

PROOF. The only links which might be created by an erasing reduction step are $?w$-links (see Definition 2.14): t' and $\overleftarrow{t'}$ might have a different type only if t' is erasing. □

Lemma 4.4 (Postponement of $\xrightarrow{\mathsf{e}}$). *The reduction $\xrightarrow{\mathsf{e}}$ can be postponed w.r.t. the reduction $\xrightarrow{\neg\mathsf{e}}$ (see Figure 4(e)).*

PROOF. Suppose $\pi \xrightarrow{\mathsf{e}} \pi_1 \xrightarrow{\neg\mathsf{e}} \pi_2$. Let u (resp. t) be the cut reduced in $\pi \xrightarrow{\mathsf{e}} \pi_1$ (resp. in $\pi_1 \xrightarrow{\neg\mathsf{e}} \pi_2$). By Lemma 4.3 and by the hypothesis that u is erasing, we deduce that t has an ancestor $\overleftarrow{t}$ in π and that t and $\overleftarrow{t}$ have the same type: in particular $\overleftarrow{t}$ is non-erasing.

We define π_3 as the result of reducing $\overleftarrow{t}$ in π (so $\pi \xrightarrow{\neg\mathsf{e}} \pi_3$). By inspection of cases one can check that reducing the residues of u (which are all still erasing) in π_3 yields π_2, i.e. $\pi_3 \xrightarrow{\mathsf{e}*} \pi_2$. □

Proposition 4.5. *Let π be an sps. If $\pi \in \mathrm{SN}^{\neg\mathsf{e}}$ then $\pi \in \mathrm{SN}$.*

PROOF. Suppose that $\pi \notin \mathrm{SN}$ and consider an infinite reduction sequence starting from π: $R := \pi \xrightarrow{\mathtt{cut}} \pi_1 \xrightarrow{\mathtt{cut}} \pi_2 \xrightarrow{\mathtt{cut}} \dots$. Observe first that R has no infinite suffix of erasing steps, since the number of links strictly decreases at each erasing step.

We will define for any number n a sequence Q of $\xrightarrow{\neg\mathsf{e}}$ steps of length n starting from π, hence proving that $\pi \notin \mathrm{SN}^{\neg\mathsf{e}}$. Let k be the least number (if any) s.t. $\pi_k \xrightarrow{\mathsf{e}} \pi_{k+1}$. We define Q by induction on $n - k$.

If $k \geq n$ or k does not exist, then simply take Q as the prefix of R of length n. If $k < n$, let m be the least integer s.t. $m > k$ and $\pi_m \xrightarrow{\neg\mathsf{e}} \pi_{m+1}$ (this m does exist since R has no infinite suffix of erasing steps). Apply $m - k$ times Lemma 4.4, for obtaining an (infinite) sequence of reductions R' which has a prefix of length $k + 1$ of $\xrightarrow{\neg\mathsf{e}}$ steps. We obtain Q by applying the induction hypothesis to R'. □

4.2. $\mathrm{SN}^{\neg\mathsf{e}}$ *is a consequence of* $\mathrm{WN}^{\neg\mathsf{e}}$

This subsection is devoted to prove Proposition 4.10: the weak and strong normalization of $\xrightarrow{\neg\mathsf{e}}$ are the same property for sps satisfying AC. Our proof is quite delicate and it is based on a confluence theorem (Theorem 4.18), following a method proposed by Gandy in the framework of Gödel's system T [5].

The basic idea is to find a measure $|\pi|_\ell$ on sps which is a natural number and to prove that:

(i) $|\pi|_\ell$ strictly increases under $\xrightarrow{\neg\mathsf{e}}$,
(ii) $\xrightarrow{\neg\mathsf{e}}$ is confluent.

Then if an sps π has a normal form π' (i.e. if $\pi \in \mathrm{WN}^{\neg\mathsf{e}}$), we can deduce that the number $|\pi'|_\ell$ maximizes the length of every $\xrightarrow{\neg\mathsf{e}}$ reduction sequence starting from π (and thus $\pi \in \mathrm{SN}^{\neg\mathsf{e}}$).

In order to define this increasing measure, we change a bit the syntax of sps, defining the labelled sps (Definition 4.6) and the labelled cut-elimination (Definition 4.7). The labelling plays the role of "counting" the number of steps applied to an sps, thus allowing the definiton of an increasing measure (Definition 4.8).

Our guidelines in the definition of labelled sps and of their cut-elimination are the following:

(i) labels will be used to define the measure which has to increase under cut-elimination
(ii) to apply Gandy's method we need confluence for *labelled* sps.

We then introduce "new" links (actually dummy links) whose unique role is to participate (by means of their labels) to the measure of the sps: we call them $*$-links. These dummy links are the only labelled ones. The idea is to associate with every flat of an sps a unique $*$-link, whose label is actually "the label of the flat".

Definition 4.6 (Labelled sps). The $*$-*link* is a link without premises nor conclusions. A *labelled sliced pure structure*, s$^\ell$ps for short, is a couple $\langle \pi, \ell \rangle$ s.t. π is an sps with exactly one occurrence of the $*$-link in every flat of π, and ℓ is a function from the occurrences of $*$-links of π to the natural numbers, which is the *labelling of* $\langle \pi, \ell \rangle$.

The *degree of the labelling of* $\langle \pi, \ell \rangle$, denoted by $|\ell|$, is the sum of ℓ's values on the occurrences of $*$-links of π[25].

Definition 4.7 (Cut elimination for s$^\ell$ps). Let t be a cut of $\langle \pi, \ell \rangle$. We allow to reduce t only in case it is non erasing. The result of the reduction of t is the s$^\ell$ps $\langle \pi', \ell' \rangle$ defined as follows:

1. π' is the result of the reduction of t in π defined as in Definition 2.12, except if t is of type $(!/?d)$: in that case, first erase the $*$-link of the flat of t and then proceed like in Definition 2.12; in such a way, every flat of π' contains exactly one $*$-link;
2. let r be a $*$-link of π', we define $\ell'(r)$. Let $\overleftarrow{r}$ be the ancestor of r in π[26]. There are three possible cases:
   (a) in case t is of type (ax), $(1/\bot)$, $(\otimes/\parr)$ or $(\oplus_i/\&_i)$, if the flat of $\overleftarrow{r}$ is the same as the flat of t, set $\ell'(r) = \ell(\overleftarrow{r}) + 1$; otherwise set $\ell'(r) = \ell(\overleftarrow{r})$;
   (b) in case t is of type $(!/?d)$, let v (resp. o) be the $*$-link (resp. !-link) in π erased by the reduction of t. If $\overleftarrow{r}$ has depth 0 in the sps associated with o, then set $\ell'(r) = \ell(\overleftarrow{r}) + \ell(v) + 2$; otherwise set $\ell'(r) = \ell(\overleftarrow{r})$;
   (c) in case t is of type $(!/?c)$ or $(!/!)$, set $\ell'(r) = \ell(\overleftarrow{r})$.

We introduce the notation $t(\langle \pi, \ell \rangle)$ and $\xrightarrow{\ell}$, as usual.

Definition 4.8 (ℓ-measure). Let $\langle \pi, \ell \rangle$ be an s$^\ell$ps, c be the number of !-links of π, d be the sum of the depths of the !-links of $\langle \pi, \ell \rangle$. The ℓ-*measure of* $\langle \pi, \ell \rangle$ is a natural number, denoted by $| \langle \pi, \ell \rangle |_\ell$ and defined as follows:

$$| \langle \pi, \ell \rangle |_\ell \quad := \quad (|\ell| + c)^2 + d$$

[25] Notice that with two different occurrences of the same flat of $\langle \pi, \ell \rangle$ are associated two different occurrences of the $*$-link: this entails that our definition of $|\ell|$ has taken into account the multiplicities of a given flat in the sps associated with a given !-link. We will as usual write in the sequel "a $*$-link" always meaning "an occurrence of the $*$-link".

[26] The definition of ancestor/residue of a $*$-link is straightforward, from the definition of π' given in step 1: every $*$-link of π' has exactly one ancestor in π.

We now prove the first key property of the reduction $\xrightarrow{\ell}$, namely that it strictly increases the ℓ-measure:

Lemma 4.9. *Let $\langle \pi, \ell \rangle$ be an s$^\ell$ps: if $\langle \pi, \ell \rangle \xrightarrow{\ell} \langle \pi', \ell' \rangle$, then $| \langle \pi', \ell' \rangle |_\ell > | \langle \pi, \ell \rangle |_\ell$.*

PROOF. In case $\langle \pi', \ell' \rangle$ is the result of an (ax), $(1/\bot)$, $(\otimes/\parr)$ or $(\oplus_i/\&_i)$ step, then $|\ell'| = |\ell| + 1$, $c' = c$, $d' = d$, hence $| \langle \pi', \ell' \rangle |_\ell > | \langle \pi, \ell \rangle |_\ell$.
In case $\langle \pi', \ell' \rangle$ is the result of a $(!/?c)$ step, then $|\ell'| \geq |\ell|$, $c' \geq c + 1$, $d' \geq d$, hence $| \langle \pi', \ell' \rangle |_\ell > | \langle \pi, \ell \rangle |_\ell$.
In case $\langle \pi', \ell' \rangle$ is the result of a $(!/!)$ step, then $|\ell'| \geq |\ell|$, $c' \geq c$, $d' \geq d + 1$, hence $| \langle \pi', \ell' \rangle |_\ell > | \langle \pi, \ell \rangle |_\ell$.
In case $\langle \pi', \ell' \rangle$ is the result of a $(!/?d)$ step, then $|\ell'| \geq |\ell|+2$, $c' \geq c-1$, $d' \geq d-c+1$.
We deduce: $| \langle \pi', \ell' \rangle |_\ell \geq (|\ell| + c + 1)^2 + d - c + 1 \geq | \langle \pi, \ell \rangle |_\ell + c + 1$. □

The second key property of the reduction $\xrightarrow{\ell}$ is confluence (Thereom 4.18), which is proven in the next subsection. We prove now that the two mentioned key properties of $\xrightarrow{\ell}$ entail the equivalence of $\mathrm{WN}^{\neg\mathsf{e}}$ and $\mathrm{SN}^{\neg\mathsf{e}}$ for sps satisfying AC:

Proposition 4.10. *Let π be an sps which satisfies* AC. *If $\pi \in \mathrm{WN}^{\neg\mathsf{e}}$ then $\pi \in \mathrm{SN}^{\neg\mathsf{e}}$.*

PROOF. Suppose π is an sps satisfying AC and s.t. $\pi \in \mathrm{WN}^{\neg\mathsf{e}}$. For every ℓ, $\langle \pi, \ell \rangle$ satisfies AC and $\langle \pi, \ell \rangle \in \mathrm{WN}^\ell$. We prove that $\langle \pi, \ell \rangle \in \mathrm{SN}^\ell$, which clearly implies that $\pi \in \mathrm{SN}^{\neg\mathsf{e}}$.

Let $\langle \pi', \ell' \rangle$ be a normal form of $\langle \pi, \ell \rangle$: we prove that the length of every reduction sequence starting from $\langle \pi, \ell \rangle$ is bounded by $| \langle \pi', \ell' \rangle |_\ell$.

Suppose $\langle \pi, \ell \rangle \xrightarrow{\ell} \langle \pi_1, \ell_1 \rangle \xrightarrow{\ell} \dots \xrightarrow{\ell} \langle \pi_n, \ell_n \rangle$. By Lemma 4.9 we have $| \langle \pi_n, \ell_n \rangle |_\ell \geq | \langle \pi, \ell \rangle |_\ell + n$. Since $\langle \pi, \ell \rangle$ satisfies AC we deduce by confluence (Theorem 4.18) that $\langle \pi_n, \ell_n \rangle \xrightarrow{\ell*} \langle \pi', \ell' \rangle$, hence by Lemma 4.9, $| \langle \pi', \ell' \rangle |_\ell \geq | \langle \pi_n, \ell_n \rangle |_\ell$. We then conclude that $| \langle \pi', \ell' \rangle |_\ell \geq n$. □

4.2.1. Confluence of $\xrightarrow{\ell}$

The rest of this section is devoted to the proof of confluence of ℓ-reduction (Theorem 4.18). Our ℓ-reduction (as well as usual LL cut-reduction) is locally confluent but not strongly confluent (recall Figure 4(c)). The reader can find counter-examples to the strong confluence property in the proofs of Lemma 4.12 and Lemma 4.14.

We thus prove the confluence of ℓ-reduction (Theorem 4.18) by decomposing $\xrightarrow{\ell}$ into its logical and structural subreduction, $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$, exactly as we did for $\xrightarrow{\mathtt{cut}}$ in Definition 3.1. Then we show that both $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$ are confluent (Proposition 4.13 and Proposition 4.15) and that they commute (Lemma 4.17). We conclude that $\xrightarrow{\ell}$ is confluent since it is the union of two confluent reductions which commute (Hindley-Rosen Lemma, here Lemma 2.8).

Definition 4.11. The *ℓ-logical reduction*, denoted by $\xrightarrow{\mathtt{log}\ell}$, is the context closure of the following ℓ-reduction steps: (ax), $(\otimes/\parr)$, $(1/\bot)$, $(\oplus_i/\&_i)$, $(!/?d)$; the *ℓ-structural reduction*, denoted by $\xrightarrow{\mathtt{str}\ell}$, is the context closure of the following ℓ-reduction steps: $(!/!)$ and $(!/?c)$.

Of course we have $\xrightarrow{\ell} = \xrightarrow{\mathtt{log}\ell} \cup \xrightarrow{\mathtt{str}\ell}$. Notice that $\xrightarrow{\mathtt{str}\ell}$ is defined precisely by those ℓ-steps which leave unchanged the labels of the $*$-links.

In what follows we prove that $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$ are confluent and commute. The difficult part of the proof is already achieved: it consists in establishing that both $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$ are SN, which is an immediate consequence of SN of $\xrightarrow{\mathtt{log}}$ (Proposition 3.4) and of $\xrightarrow{\mathtt{str}}$ (Proposition 4.13): the labelling of s$^\ell$ps plays no role w.r.t. SN.

Confluence of $\xrightarrow{\mathtt{log}\ell}$. We prove the confluence of $\xrightarrow{\mathtt{log}\ell}$ (Proposition 4.13) by applying the Newman Lemma (here Lemma 2.7): a relation which is locally confluent and SN is confluent. The local confluence of $\xrightarrow{\mathtt{str}\ell}$ is proven by Lemma 4.12, and the SN of $\xrightarrow{\mathtt{log}\ell}$ is an immediate consequence of Proposition 3.4.

Neither in the proof of Proposition 3.4 nor in the one of Lemma 4.12 the AC condition is used, so that the confluence of $\xrightarrow{\mathtt{log}\ell}$ is established for the whole set of s$^\ell$ps.

Lemma 4.12. *The reduction* $\xrightarrow{\mathtt{log}\ell}$ *is locally confluent on s$^\ell$ps.*

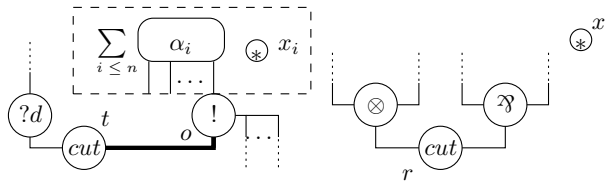
PROOF. We prove that for every slice $\langle \alpha, \ell \rangle$ the following diagram holds:

$$\begin{array}{ccc} \langle \alpha, \ell \rangle & \xrightarrow{\mathtt{log}\ell} & \langle \pi_2, \ell_2 \rangle \\ \downarrow{\scriptstyle \mathtt{log}\ell} & & \vdots{\scriptstyle \mathtt{log}\ell *} \\ \langle \pi_1, \ell_1 \rangle & \overset{\mathtt{log}\ell *}{\cdots\cdots\cdots} & \langle \pi_3, \ell_3 \rangle \end{array}$$

This immediately entails local confluence for general s$^\ell$ps.

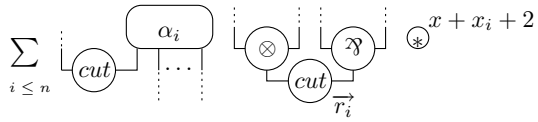
Establishing that the previous diagram holds is not immediate only when at least one reduction is of type $(!/?d)$: in this case[27] the slice $\langle \alpha, \ell \rangle$ is duplicated a number of times equal to the number of slices of the box opened by the $(!/?d)$ reduction. Let us consider for example the case $\langle \alpha, \ell \rangle \xrightarrow{\mathtt{log}} \langle \pi_1, \ell_1 \rangle$ is a $(!/?d)$ step and $\langle \alpha, \ell \rangle \xrightarrow{\mathtt{log}} \langle \pi_2, \ell_2 \rangle$ is a $(\otimes/\parr)$ step (the other cases are similar and left to the reader). We have, for some $n > 0$:

$\langle \alpha, \ell \rangle =$

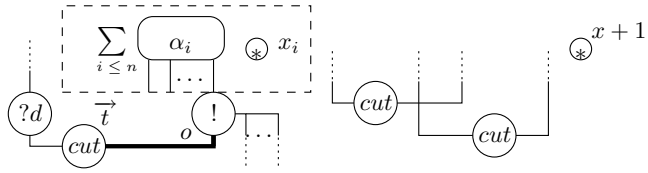


the reduction of the cut t opens the box of the !-link o and duplicates the slice $\langle \alpha, \ell \rangle$ n times, giving as result the following s$^\ell$ps (notice that the labelling of the $*$-links changes):

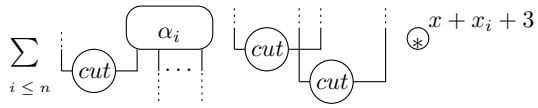
[27] We can suppose without loss of generality that the reduced cut link has depth 0 in α.

$\langle \pi_1, \ell_1 \rangle =$ 

Instead, by reducting the cut r in $\langle \alpha, \ell \rangle$, one obtains the following slice:

$\langle \pi_2, \ell_2 \rangle =$ 

If we reduce the cut $\overrightarrow{t}$ in $\langle \pi_2, \ell_2 \rangle$, we obtain the following s$^\ell$ps (again notice that the labelling of the $*$-links changes):

$\langle \pi_3, \ell_3 \rangle =$ 

The same s$^\ell$ps can be obtained by reducing the residues $\overrightarrow{r_1}, \ldots, \overrightarrow{r_n}$ of the cut r in $\langle \pi_1, \ell_1 \rangle$. We conclude that $\langle \pi_1, \ell_1 \rangle \xrightarrow{\texttt{log}\ell}{}^{*} \langle \pi_3, \ell_3 \rangle$ and $\langle \pi_2, \ell_2 \rangle \xrightarrow{\texttt{log}\ell} \langle \pi_3, \ell_3 \rangle$. □

Notice that the critical pair analysed in the proof of Lemma 4.12 is a counter-example to the strong confluence of $\xrightarrow{\texttt{log}\ell}$.

Proposition 4.13. *The reduction $\xrightarrow{\texttt{log}\ell}$ is confluent on the s$^\ell$ps.*

PROOF. Consequence of Lemma 4.12, Proposition 3.4 and the Newman Lemma (here Lemma 2.7). □

Confluence of $\xrightarrow{\texttt{str}\ell}$. As for $\xrightarrow{\texttt{log}\ell}$, the confluence of $\xrightarrow{\texttt{str}\ell}$ (Proposition 4.15) is obtained using the Newman Lemma. The local confluence is deduced by Lemma 4.14, and the SN of $\xrightarrow{\texttt{str}\ell}$ is an immediate consequence of Proposition 3.10. In sharp contrast with the case of the $\xrightarrow{\texttt{log}\ell}$ rewriting rule, the reader should remark that the AC condition plays a crucial role both for Lemma 4.14 and Proposition 3.10: in Subsection 2.4 we have given counter-examples (see Fig. 11 and Fig. 12) both to the local confluence and the SN of $\xrightarrow{\texttt{str}}$ for sps which do not satisfy AC.

Remark that the labelling of the $*$-links does not play any role in the confluence of $\xrightarrow{\texttt{str}\ell}$, since the ℓ function is invariant under $\xrightarrow{\texttt{str}\ell}$. We nevertheless picture explicitely the $*$-links in the figures illustrating the following lemma, in order to stress that confluence holds for *labelled* sps.

Lemma 4.14. *The reduction* $\xrightarrow{\mathtt{str}\ell}$ *is locally confluent on the* s$^\ell$*ps satisfying* AC.

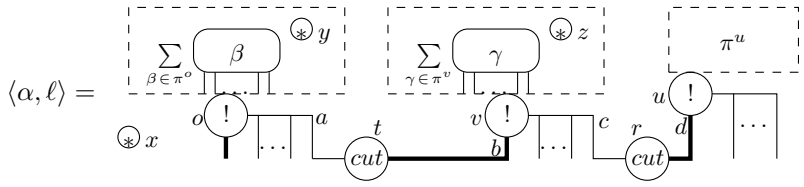
PROOF. The local confluence of $\xrightarrow{\mathtt{str}\ell}$ is quite a standard result, the proof is essentially the same as that of the local confluence of the exponential reduction in MELL given by V. Danos in his PhD thesis [2]. The local confluence of $\xrightarrow{\mathtt{str}\ell}$ can be reduced to the following diagram:

$$\begin{array}{ccc} \langle \alpha, \ell \rangle & \xrightarrow{\mathtt{str}\ell} & \langle \pi_2, \ell_2 \rangle \\ \downarrow{\scriptstyle \mathtt{str}\ell} & & \vdots{\scriptstyle \mathtt{str}\ell *} \\ \langle \pi_1, \ell_1 \rangle & \xrightarrow{\mathtt{str}\ell *} & \langle \pi_3, \ell_3 \rangle \end{array}$$

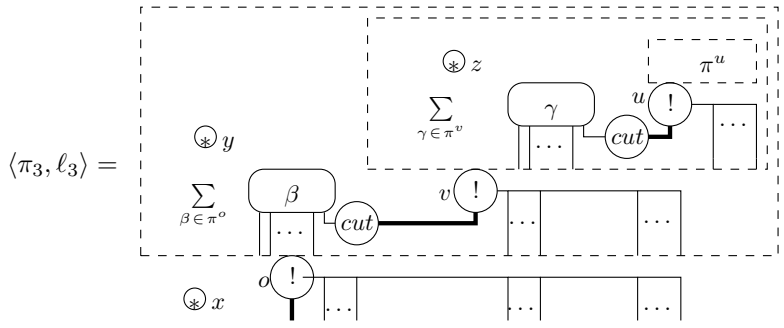
This immediately entails local confluence for general s$^\ell$ps.

The proof is by inspection of all possible cases, we treat in detail only the following two critical pairs (the considered cuts have depth 0 in α).

1. If $\langle \alpha, \ell \rangle \xrightarrow{\mathtt{str}\ell} \langle \pi_1, \ell_1 \rangle$ is the reduction of a cut t of type $(!/!)$ and $\langle \alpha, \ell \rangle \xrightarrow{\mathtt{str}} \langle \pi_2, \ell_2 \rangle$ is the reduction of a cut r of type $(!/!)$ and if the two cuts are in opposition as pictured below:

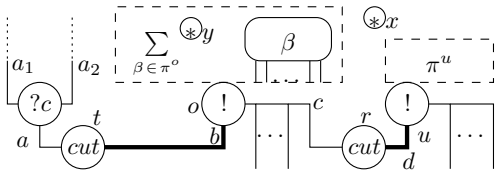


then the reduction of t will bring a copy of v in each slice β of the s$^\ell$ps associated with o and it will transform the premise c of the cut r in an auxiliary conclusion of the residue of the !-link o in $\langle \pi_1, \ell_1 \rangle$. On the other hand the reduction of r in $\langle \alpha, \ell \rangle$ will bring a copy of u in each slice γ of the s$^\ell$ps associated with v. The s$^\ell$ps $\langle \pi_3, \ell_3 \rangle$ is obtained from $\langle \pi_2, \ell_2 \rangle$ by reducing the (unique) residue of t in $\langle \pi_2, \ell_2 \rangle$, which results in bringing v, and hence u, in each slice of the s$^\ell$ps associated with o:

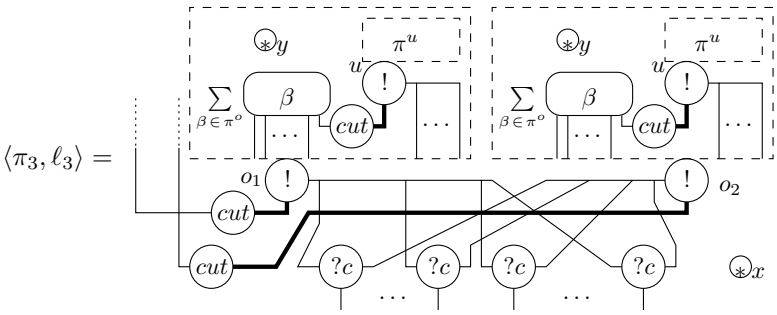


The s$^\ell$ps $\langle \pi_3, \ell_3 \rangle$ can also be obtained from $\langle \pi_1, \ell_1 \rangle$ in two steps (recall we are proving only local confluence of $\xrightarrow{\texttt{str}\ell}$): first, we reduce the residue of r in $\langle \pi_1, \ell_1 \rangle$ thus obtaining for each slice β in the box of o a cut between an auxiliary conclusion of a copy of v and a copy of u, and then we reduce every such cut, thus bringing a copy of u in every slice γ in the boxes of the various copies of v, as sketched in the above figure.

2. If $\langle \alpha, \ell \rangle \xrightarrow{\texttt{str}\ell} \langle \pi_1, \ell_1 \rangle$ is the reduction of a cut t of type $(!/?c)$ and $\langle \alpha, \ell \rangle \xrightarrow{\texttt{str}\ell} \langle \pi_2, \ell_2 \rangle$ is the reduction of a cut r of type $(!/!)$ and if the two cuts are in opposition as pictured below:

$\langle \alpha, \ell \rangle =$ 

then the reduction of t will duplicate o in $\langle \pi_1, \ell_1 \rangle$ and will transform the cut r in a cut of type $(!/?c)$ between the residue of the !-link u and a created $?c$-link with premises the auxiliary conclusions corresponding to c of the two copies of o. On the other hand, the reduction of r in $\langle \alpha, \ell \rangle$ will bring a copy of u in each slice β of the s$^\ell$ps associated with o. The s$^\ell$ps $\langle \pi_3, \ell_3 \rangle$ is obtained from $\langle \pi_2, \ell_2 \rangle$ by reducing the (unique) residue of t in $\langle \pi_2, \ell_2 \rangle$, which duplicates o, and hence u:

$\langle \pi_3, \ell_3 \rangle =$ 

The s$^\ell$ps $\langle \pi_3, \ell_3 \rangle$ can also be obtained from $\langle \pi_1, \ell_1 \rangle$ in three steps (also in this case we have only local confluence and not strong confluence): first, we reduce the residue of r in $\langle \pi_1, \ell_1 \rangle$, which is a cut of type $(!/?c)$; this duplicates u and creates two cuts of type $(!/!)$, one between the first copies of o and u and the other one between the second copies of o and u. We obtain $\langle \pi_3, \ell_3 \rangle$ by further reducing the two $(!/!)$ cuts.

□

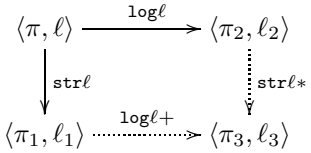
The critical pairs treated in the proof of Lemma 4.14 are counter-examples to the strong confluence of $\xrightarrow{\mathtt{str}\ell}$.

Proposition 4.15. *The reduction $\xrightarrow{\mathtt{str}\ell}$ is confluent on the s$^\ell$ps satisfying* AC.

PROOF. Consequence of Lemma 4.14, Proposition 3.10 and the Newman Lemma (Lemma 2.7). □

Commutation of $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$. The last step allows to merge $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$ together, proving that the two reductions commute (Lemma 4.17): this is achieved by applying a lemma by Di Cosmo, Piperno and Geser (here Lemma 2.9) which reduces the commutation of $\xrightarrow{\mathtt{log}\ell}$ and $\xrightarrow{\mathtt{str}\ell}$ to the diagram (1) of Lemma 2.9.
Like in the proof of confluence for $\xrightarrow{\mathtt{log}\ell}$ (and contrary to the proof of confluence for $\xrightarrow{\mathtt{str}\ell}$) the AC condition is not needed in this paragraph.

Lemma 4.16. *For every s$^\ell$ps the following diagram holds:*



PROOF. Also for this lemma we restrict ourselves to the case $\langle \pi, \ell \rangle$ is a slice $\langle \alpha, \ell \rangle$: the general case follows immediately.

Let t (resp. r) be the cut link of α reduced in $\langle \alpha, \ell \rangle \xrightarrow{\mathtt{str}\ell} \langle \pi_1, \ell_1 \rangle$ (resp. in $\langle \alpha, \ell \rangle \xrightarrow{\mathtt{log}\ell} \langle \pi_2, \ell_2 \rangle$). The proof is by induction on the depth of r. We split the proof in three cases.

Case (i). The cut r is at depth 0 in $\langle \alpha, \ell \rangle$: this case is immediate, since the reduction of t does not affect r. The only slightly delicate case is when r is of type $(!/?d)$ (so that its reduction duplicates $\langle \alpha, \ell \rangle$) and t is duplicated a number of times equal to the number of slices, say n, of the opened exponential box: $\pi_2 = \sum_{i=1}^{n} \alpha_i$. Since r is a non erasing cut link, we have $n \geq 1$. In particular, let $\overrightarrow{t_1}, \ldots, \overrightarrow{t_n}$ be the n residues of t in $\langle \pi_2, \ell_2 \rangle$. We first prove that the diagram holds without labels, by distingushing two subcases: (i) if t is of type $(!/!)$ and one of the two premises of t is an auxiliary conclusion of the !-link opened by the reduction of r, then we do not need to reduce the residues of t to close the diagram: $\pi_3 = \pi_2$ [28]; (ii) otherwise the reduction of the residues of t in π_2 gives as result the same sps as the one given by the reduction of the (unique) residue of r in π_1. Notice that π_3 has exactly n slices: $\pi_3 = \sum_{i=1}^{n} \beta_i$.

Concerning the labels, let's call v_i the $*$-link at depth 0 of the slice α_i of π_2 and v the $*$-link of α. Similarly, let us call v'_i the $*$-link at depth 0 of the slice β_i of π_3. The

[28] Notice that in this case (and only in this case) these t's residues might also be non reducible cuts.

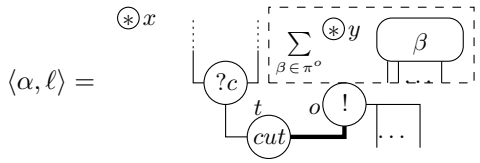
computation of the label of v_i' yields (for both the considered reductions from π to π_3) $\ell(v) + \ell(\overleftarrow{v_i}) + 2$. Since the label of every $*$-link of π_3 different from v_i' is the same as the one of its ancestor in π, the diagram holds also with labels.

Case (ii). Both t and r are cuts in boxes of α. Let o (resp. u) be the !-link at depth 0 of α containing t (resp. r). If o and u are different !-links, then this case is immediate; if they are the same !-link o, let $\langle \pi^o, \ell^o \rangle$ be the box associated with o. We have $\langle \pi^o, \ell^o \rangle \xrightarrow{\mathtt{str}\ell} \langle \pi_1^o, \ell_1^o \rangle$ and $\langle \pi^o, \ell^o \rangle \xrightarrow{\mathtt{log}\ell} \langle \pi_2^o, \ell_2^o \rangle$. We apply the induction hypothesis and we obtain an s$^\ell$ps $\langle \pi_3^o, \ell_3^o \rangle$ s.t. $\langle \pi_1^o, \ell_1^o \rangle \xrightarrow{\mathtt{log}\ell+} \langle \pi_3^o, \ell_3^o \rangle$ and $\langle \pi_2^o, \ell_2^o \rangle \xrightarrow{\mathtt{str}\ell*} \langle \pi_3^o, \ell_3^o \rangle$. The s$^\ell$ps $\langle \pi_3, \ell_3 \rangle$ is then obtained from $\langle \pi, \ell \rangle$ by associating with o the s$^\ell$ps $\langle \pi_3^o, \ell_3^o \rangle$.

Case (iii). The cut t is at depth 0 and r is contained in a !-link o at depth 0 of α: then we can have two critical pairs, since the !-link o may be involved in the reduction of t.

One critical pair is when t is of type $(!/?c)$ and o is the !-link which is duplicated by the reduction of t:

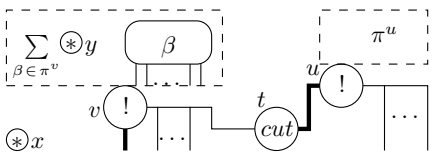
$\langle \alpha, \ell \rangle =$



The reduction of t duplicates o, hence r, in $\langle \pi_1, \ell_1 \rangle$, but does not change the content of the box nor the labels of the $*$-links. On the other hand, the reduction of r changes the s$^\ell$ps associated with o, without affecting the cut t nor the label of the $*$-link at depth 0 of α, but possibly changing the labels of the $*$-links inside o. Reducing the two residues of r in $\langle \pi_1, \ell_1 \rangle$ gives the same result $\langle \pi_3, \ell_3 \rangle$ as reducing the residue of t in $\langle \pi_2, \ell_2 \rangle$. By the way notice that the reduction $\langle \pi_1, \ell_1 \rangle \xrightarrow{\mathtt{log}\ell+} \langle \pi_3, \ell_3 \rangle$ costs two steps, due to the duplication of r.

The other critical pair is when t is of type $(!/!)$ and o (the !-link whose box contains r) is one of the two !-links involved in the reduction of t (i.e. in the figure below $o = u$ or $o = v$):

$\langle \alpha, \ell \rangle =$



The reduction of t brings one copy of u in each slice of the box associated with v (notice that v contains at least one slice, since $\xrightarrow{\mathtt{str}\ell}$ is a non-erasing reduction step): however, this reduction does not change the s$^\ell$ps associated with u nor the labels of the $*$-links. On the other hand the reduction of r in $\langle \alpha, \ell \rangle$ changes the s$^\ell$ps associated with u or that associated with v, depending on whether $o = u$ or $o = v$.

The case $o = u$ is simple: let $\overrightarrow{r}_1, \ldots, \overrightarrow{r}_n$ (where $n \geq 1$, as we noticed above) be the residues of r in $\langle \pi_1, \ell_1 \rangle$. Reducing these n cuts gives the same result as reducing the unique residue of t in $\langle \pi_2, \ell_2 \rangle$.

If $o = v$, then one has to notice that even if the reduction of t changes the box π^v associated with v (bringing "inside v" the !-link u), cuts and $*$-links of π^v are not affected by this reduction. In particular we can reduce the residue of r in $\langle \pi_1, \ell_1 \rangle$ like the reduction step $\langle \alpha, \ell \rangle \xrightarrow{\texttt{log}\ell} \langle \pi_2, \ell_2 \rangle$ does, thus obtaining the s$^\ell$ps $\langle \pi_3, \ell_3 \rangle$ which is also the result of reducing the residue of t in $\langle \pi_2, \ell_2 \rangle$. □

Lemma 4.17. *The reductions* $\xrightarrow{\texttt{str}\ell}$ *and* $\xrightarrow{\texttt{log}\ell}$ *commute (see Figure 4(d)).*

PROOF. It is a consequence of Lemma 4.16, Prop. 3.4 and a Lemma by Di Cosmo, Piperno and Geser (here Lemma 2.9). □

Theorem 4.18. *The reduction* $\xrightarrow{\ell}$ *is confluent on s*$^\ell$*ps satisfying* AC.

PROOF. Consequence of Prop. 4.13, Prop. 4.15, Lemma 4.17 and the Hindley-Rosen Lemma (Lemma 2.8). □

Remark 4.19. One can easily adapt Theorem 4.18 and prove the confluence of $\xrightarrow{\texttt{cut}}$ for $\top$-free sps satisying AC: just check that $\xrightarrow{\texttt{log}}$ and $\xrightarrow{\texttt{str}}$ are locally confluent (i.e. add the erasing steps to the proof of Lemma 4.12 and Lemma 4.14) and that they commute.

$$F := \begin{array}{ccccccccccccc} a & | & 1 & | & F \otimes F & | & 0 & | & F \oplus F & | & !F & | & \exists a.F \\ a^{\perp} & | & \perp & | & F \parr F & | & \top & | & F \,\&\, F & | & ?F & | & \forall a.F \end{array}$$

Figure 17: Grammar of LL formulas

5. Strong Normalization for Linear Logic

We now want to apply our (rather general) main result (Theorem 4.2) in order to prove strong normalization for full second order LL (Theorem 5.12). As proof-nets, we take here the most general currently used notion, obtained by combining [7], [9] and [2], and by generalizing the (ccad) and the $(\top/\mathrm{cc})$ reduction steps (see below). These proof-nets are defined in [24]: let's call them simply *nets*, and denote them with initial Greek letters $\alpha, \beta \ldots$

This last section requires some knowledge on nets, mainly some acquaintance with the additive and second order cut-elimination and the notion of sequentialization: all the precise definitions are in [24].

5.1. The syntax of nets

There are two main differences between nets and sps: (i) nets are typed by second order LL formulas (Figure 17 recalls the grammar of LL) and (ii) nets use additive boxes to handle the $\&$ rule. Let us comment this last feature.

In the framework of nets, $\&$-links behave like !-links (see Figure 18): they have 0 premises and $n+1$ conclusions, a distinguished one (the main conclusion, typed by a $\&$-formula, say $A \,\&\, B$) and possibly others (the auxiliary conclusions); with every $\&$-link with $n+1$ conclusions are associated two nets β_1 and β_2, called *first (or left)* and *second (or right)* component of the $\&$-link. The nets β_1 and β_2 have the same n conclusions as the auxiliary conclusions of the $\&$-link (called the auxiliary conclusions of the component of the $\&$-link) and one distinguished conclusion (of type respectively A and B, and called the main conclusion of the component of the $\&$-link).

The presence of additive boxes has two main consequences in the definition of the cut-elimination steps: (1) there is a unique additive step $(\&/\oplus_i)$ which combines an erasing feature (like the step $(\&_i/\oplus_j)$ with $i \neq j$, in sps) and a non-erasing feature (like the step $(\&_i/\oplus_i)$ in sps); (2) it yields a new type of reduction step, called (ccad), which

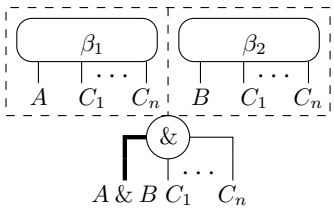


Figure 18: An example of additive box

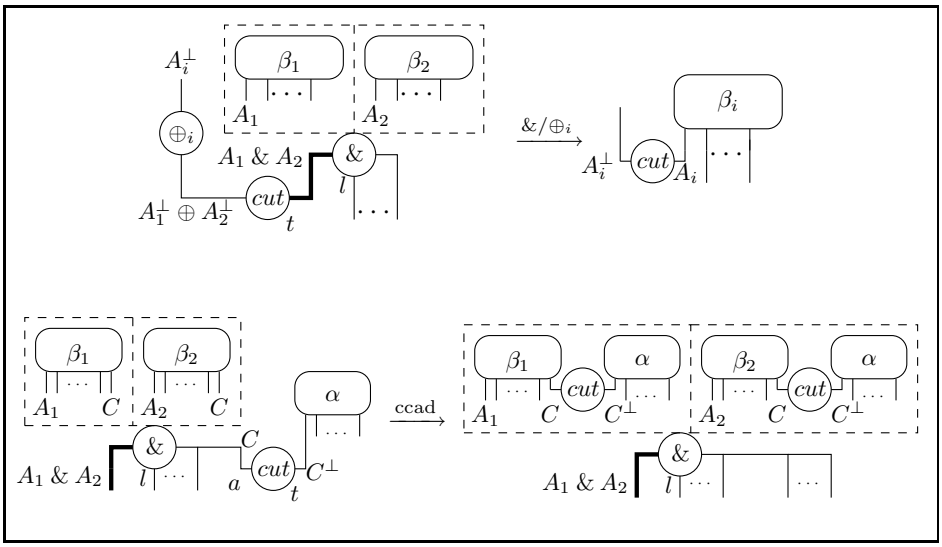
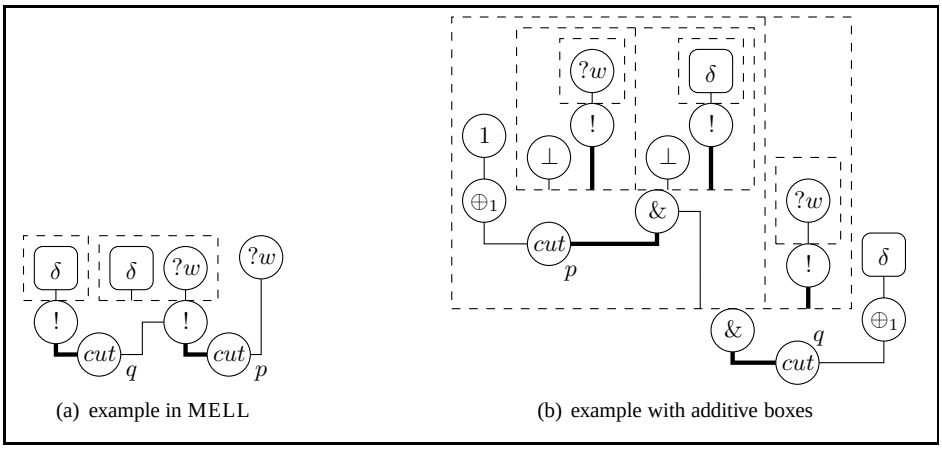


Figure 19: The reduction steps associated with the additive box

is the nightmare of this way of representing proofs (the same holds for the $(\top/\mathrm{cc})$ step of Definition 2.12).

The step $(\&/\oplus_i)$**.** A redex of type $(\&/\oplus_i)$ is made of a cut t, a $\&$-link and a $\oplus_i$-link such that one premise of t is the main conclusion of the $\&$-link and the other premise of t is the conclusion of the $\oplus_i$-link (see Figure 19). The contractum is defined by erasing the $\oplus_i$-link, by substituting the $\&$-link with its i^{th} component and cutting the premise of the erased $\oplus_i$-link and the main conclusion of the i^{th} component of the $\&$-link. As already mentioned, this reduction step is both erasing (it erases one component of the $\&$-link) and non-erasing (it modifies the cut formula like for example the $(\otimes/\parr)$ step). This mix is critical with respect to standardization, as we will discuss in details in the next Subsection 5.2.

The step (ccad)**.** A redex of type (ccad) is made of a cut t, a $\&$-link l and a net α not containing l; moreover, one premise of t, call it a, is an auxiliary conclusion of l, and the other premise of t is conclusion of α (see Figure 19). The contractum is defined as follows. We substitute l, the cut link t and α by a new $\&$-link (which we still call l), having the same conclusions as the original l where we have substituted the edge a by the conclusions of α (different from t's premise). The i^{th} component of the new link l is obtained by cutting the conclusion corresponding to the edge a of the i^{th} component of the original l and the conclusion of α which is a premise of t. Notice that a cut of type $(\texttt{ccad})$ of a net β can be associated with different redexes, since it is not clear which subnet α of β should be selected (for example, in [7] α is the *empire of* a, in [24] α is any subnet having a among its conclusions). We take here the (more general) option of [24], which is also applied to the $(\top/\mathrm{cc})$ step. Notice that, as for $(\top/\mathrm{cc})$ (Remark 2.13), the presence of (ccad)-reduction steps entails the failure of confluence.



(a) example in MELL (b) example with additive boxes

Figure 20: Untyped nets which are WN but not SN (δ is defined in Figure 9)

Despite these differences, the cut-elimination procedure for sps is the key tool to prove SN for nets. We associate (Subsection 5.3) with every net β its set of slices $\mathrm{sl}(\beta)$ (which is an sps), and we prove that $\beta \in \mathrm{SN}$ by proving that $\mathrm{sl}(\beta) \in \mathrm{SN}$. More precisely, we prove that for every net β:

(i) $\mathrm{sl}(\beta)$ satisfies AC,
(ii) $\beta \in \mathrm{SN}$ if $\mathrm{sl}(\beta) \in \mathrm{SN}$,
(iii) $\mathrm{sl}(\beta) \in \mathrm{SN}$ if $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$,
(iv) $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$.

Point (i) is trivial (Proposition 5.1); point (ii) is Proposition 5.6; point (iii) is a consequence of point (i) and of our standardization theorem (Theorem 4.2), this is the real missing point in Girard's original proof; point (iv) is the "difficult" part of the SN proof (which is outside Peano arithmetic): one uses Girard's reducibility candidates to prove $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$. It is well known that this kind of tool is very powerful and can be adapted to a lot of different situations. Indeed, Girard's proof of [7] works perfectly well if one substitutes "strong normalization" with "weak normalization", and the reader acquainted with reducibility candidates is probably already convinced that the changes needed to prove $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$ (instead of $\beta \in \mathrm{WN}$) present no major difficulty. We nevertheless give the precise definition of reducibility candidate (Definition 5.10) that we need in order to prove Theorem 5.11 (that is point (iv)), thus concluding with the strong normalization theorem for second order LL nets (Theorem 5.12).

At first sight, an alternative solution was to use Theorem 4.2 in order to prove the original Girard's standardization theorem (Theorem 4.25 p.72 of [7]). We explain in the next Subsection 5.2 why this alternative fails.

5.2. A digression on standardization

Before proving the points (i)-(iv) stated in Subsection 5.1, let us discuss the alternative solution mentioned, namely to use Theorem 4.2 in order to prove the original standardization theorem (Theorem 4.25 p.72 of [7]). We claim that this approach fails, since the original standardization is based on a "wrong" definition of standard reduction. To help the reader, we reproduce verbatim Definition 4.24 and Theorem 4.25 p.72 of [7].

> 4.24. Definition. A contraction is *standard* when it does not erase any symbol CUT besides the one explicitly considered. Concretely, this means that some parts of the configuration we replace have to be cut-free, namely:
>
> - in $(\&/\oplus_1 - SC)$, β_2 must be cut-free,
> - in $(\&/\oplus_2 - SC)$, β_1 must be cut-free,
> - in $(!/?w - SC)$, β_1 must be cut-free,
> - in $(\top - SC)$, $eA^{\perp}$ must be cut-free.
>
> A reduction is standard when made of standard contractions.

> 4.25 Theorem (Standardization Lemma). *Let β be a proof-net and assume that there is a standard reduction from β to a cut-free β'. Then β is* SN.

The problem with the above definition of standard reduction (*contraction* in the language of [7]) was already pointed out in [2] in the restricted MELL framework: recall the slice $\delta\delta$ of Figure 9, which can be considered an "untyped MELL net"[29]. As we pictured in Figure 10, we have $\delta\delta \xrightarrow{\texttt{cut+}} \delta\delta$. Now consider the untyped net β of Figure 20(a). By reducing the cut p of β one obtains a strongly normalizing untyped net; however β *is not* strongly normalizing (reduce the cut q and you will get a net having the net $\delta\delta$ of Figure 9 as subnet), despite the fact that the reduction step associated with p is standard in Girard's sense (Definition 4.24 p.72 of [7]). From our point of view, this means that Theorem 4.25 p.72 of [7] is "morally wrong", even though it is "technically correct" (it deals only with typed nets, so its conclusion is true...). The solution proposed in [2] was to freeze the cut link p and consider "strict" reductions, that is the rewriting rule obtained by forbidding $(!/?w)$ reduction steps: this leads to théorème 8.31 p.64 of [2], allowing to prove SN for MELL nets.

In presence of the additives, the same phenomenon occurs, but the solution cannot be that simple, as we now explain. Consider the untyped net β of Figure 20(b). Again, by reducing the cut p one obtains a strongly normalizing untyped net; however β *is not* strongly normalizing (reduce the cut q and the (ccad) created by the reduction of q), despite the fact that the reduction step associated with p is standard in Girard's sense. Following Danos, one would be tempted to freeze p, but this wouldn't be correct. Indeed, we should then freeze all the cuts of type $(\&/\oplus_i)$, and such a freezing hides infinite reduction sequences. For instance, the untyped net β of Figure 20(b) would be

[29] In this short discussion, we use the expression "untyped net", referring to the untyped version of the nets considered in this Subsection 5.1. An untyped net is not necessarily an sps because of the presence of additive boxes.

in normal form w.r.t. the rewriting rule induced by this freezing, despite the fact that $\beta \notin \mathrm{SN}$. The difference between the reduction steps $(!/?w)$ and $(\&/\oplus_i)$ is that while the first one is purely erasing, the second one is both erasing and non erasing. A key feature of the introduction of slices is precisely to separate the erasing aspect of the $(\&/\oplus_i)$ reduction step (taken into account by the $(\&_j/\oplus_i)$ steps with $i \neq j$) from its non erasing aspect (taken into account by the $(\&_i/\oplus_i)$ steps).

For this reason the standardization theorem for full LL can be correctly stated only for sliced pure structures (as we did in Theorem 4.2) and not for nets.

Another attempt to prove SN for LL (in presence of the additives) is contained in Okada's work [18]. The method proposed is to use phase semantics, which works well for WN, but for SN the same problem as in [7] arises (the proof of the standardization theorem), and we have to mention here that (like for [7]) the way Okada argues on this point cannot be considered as convincing (the author himself uses the expression "Sketch of proof"). Indeed:

1. Okada claims that (in certain circumstances) if $\beta \xrightarrow{\mathtt{cut}} \beta'$ and the reduced cut is of type $(!/!)$ and if $\beta' \in \mathrm{SN}$, then $\beta \in \mathrm{SN}$ ("Sketch of Proof" of lemma 6.6 p.364 case (4) of [18]). Of course (like Theorem 4.25 p.72 of [7]) the statement above is true since eventually all nets turn out to be strongly normalizable. However, the author claims that it is possible to "simulate" any reduction sequence of β by a reduction sequence of β'. This is as difficult as proving standardization, and it is the motivation for théorème 8.31 p.64 of [2]: in order to solve the problem Danos had to make the first very sharp analysis of the rewriting rule induced by LL cut-elimination (in particular he had to prove confluence for MELL);
2. in presence of the additives a *very* "restricted" cut-elimination procedure is defined (see the "&-box entering rule" of p.373 of [18])[30], so that even in case the proof were considered convincing (and we believe it shouldn't be) it would still be incomplete. With respect to this restricted procedure the extension to the additives is analysed ("Sketch of Modified Proof of lemma 6.6" p.379 of [18]). Actually, Okada's restriction is a way to eliminate the intrinsic difficulty of the (ccad) reduction step: in the present paper we found a way to keep control on this step in presence of all the other LL connectives.

5.3. Slicing nets

Let us come back to the proof of SN for nets. We hinted in Subsection 5.1 that the SN of a net is related to the SN of the sps "associated with" that net. Indeed, a net β can be *sliced* in an sps $\mathrm{sl}(\beta)$ which has the same number of conclusions as β (an example is given in Figure 21). Following [7] we define $\mathrm{sl}(\beta)$ by induction on a sequentialization of β:[31]

[30] Okada's procedure does not always allow to reduce all cuts: there are proof-nets with cuts to which the procedure cannot be applied.

[31] The procedure of *slicing* is actually independent from the chosen sequentialization and it can be applied also to non-sequentializable proof-structures (see [10]).

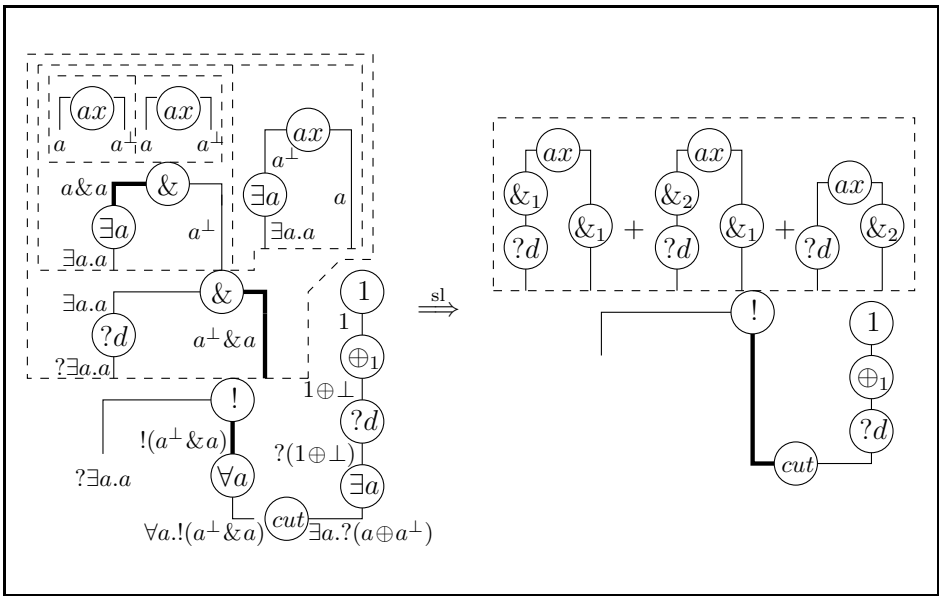


Figure 21: An example of slicing a net

- if β is an ax-link (resp. 1-link, $\top$-link), then $\mathrm{sl}(\beta)$ has only one slice, consisting of an ax-link (resp. 1-link, $\top$-link);
- if β is a $\&$-link l with associated left (resp. right) component β' (resp. β''), then $\mathrm{sl}(\beta)$ is obtained by adding a $\&_1$-link (resp. $\&_2$-link) to every slice of $\mathrm{sl}(\beta')$ (resp. $\mathrm{sl}(\beta'')$) and by taking the union of these multisets of slices (with the appropriate equivalence on the conclusions, see Definition 2.2);
- if β is a !-link l with associated net β', then $\mathrm{sl}(\beta)$ has only one slice consisting of a !-link corresponding to l and s.t. with that link is associated $\mathrm{sl}(\beta')$;
- if a conclusion of β is conclusion of a $\parr$-link (resp. of a $\bot$-, $\oplus_i$-, $?w$-, $?d$-, $?c$-link) l, let β' be the subnet of β obtained by erasing l (and its conclusion), then $\mathrm{sl}(\beta)$ is obtained by adding to every slice of $\mathrm{sl}(\beta')$ the $\parr$-link (resp. $\bot$-, $\oplus_i$-, $?w$-, $?d$-, $?c$-link) corresponding to l;
- if a conclusion of β is conclusion of a $\forall$- or $\exists$-link l, let β' be the subnet of β obtained by erasing l (and its conclusion), then $\mathrm{sl}(\beta) = \mathrm{sl}(\beta')$;
- if β is the union of two disjoint subnets β', β'' and of a $\otimes$-link (resp. cut) l, then $\mathrm{sl}(\beta)$ is obtained by connecting every slice of $\mathrm{sl}(\beta')$ and every slice of $\mathrm{sl}(\beta'')$ by means of the $\otimes$-link (resp. cut) corresponding to l (and by defining the appropriate equivalence on the conclusions, see Definition 2.2).

We first note that the slicing of a net is switching acyclic:

Proposition 5.1. *If β is a net, then $\mathrm{sl}(\beta)$ satisfies* AC.

PROOF. By the correctness criterion for nets (see [7],[24]) and the definition of sl. □

Remark 5.2. If β is a net, then by Proposition 5.1 $\mathrm{sl}(\beta)$ is deadlock free. Furthermore, a net is typed by second order LL formulas (remember Figure 17), which implies that $\mathrm{sl}(\beta)$ is also clash free. This means that every cut of $\mathrm{sl}(\beta)$ is reducible, and that $\mathrm{sl}(\beta)$ is a normal sps iff $\mathrm{sl}(\beta)$ is cut free (which is not the case for general sps).

Then we turn to the crucial point: the cut-elimination of nets can be simulated by that of its slicing, i.e. the following diagram commutes

$$\begin{array}{ccc} \beta & \xrightarrow{\texttt{cut}} & \beta' \\ \Downarrow \mathrm{sl} & & \Downarrow \mathrm{sl} \\ \mathrm{sl}(\beta) & \xrightarrow{\texttt{cut}*} & \mathrm{sl}(\beta) \end{array}$$

An example of this simulation is given in Figure 22. The reader should notice that a sequence of steps of type (ccad) or $(\forall/\exists)$ (denoted by $\xrightarrow{\mathrm{ccad}\forall/\exists *}$) is "invisible" in sps: this might be a problem to derive $\beta \in \mathrm{SN}$ from $\mathrm{sl}(\beta) \in \mathrm{SN}$, but actually it isn't thanks to Lemma 5.5.

We actually have something more than a simulation property (recall that $t(\beta')$ refers to the result of reducing a cut t in β', Definition 2.12):

Lemma 5.3 (Simulation). *Let t be a cut link of a net β. If t is of type ccad or $\forall/\exists$, then $\mathrm{sl}(\beta) = \mathrm{sl}\big(t(\beta)\big)$; otherwise, $\mathrm{sl}(\beta) \xrightarrow{+} \mathrm{sl}\big(t(\beta)\big)$.*

PROOF. Straightforward, by inspection of cases. □

We now want to prove that there is no infinite sequence of reduction steps of type (ccad) or $(\forall/\exists)$, and we use for this purpose (a straightforward variant of) the separation property proven in [24]:

Lemma 5.4. *Let β be a net. If $\beta \xrightarrow{(\mathrm{ccad})\forall/\exists *} \beta'$ and l'_1, l'_2 are two (different) residues*[32] *in β' of a $\&$-link l of β, then there exists a $\&$-link m' of β' which* separates *l'_1 and l'_2, i.e. l'_1 (resp. l'_2) is a link of the component i (resp. j) of m' and $i \neq j$ (in particular, l'_1 cannot be a link of a component of l'_2, nor the converse).*

PROOF. By induction on the length of the reduction sequence $\beta \xrightarrow{(\mathrm{ccad})\forall/\exists *} \beta'$, see [24]. □

Lemma 5.5. *There is no infinite sequence of reduction steps of type (ccad) or $(\forall/\exists)$ starting from a net.*

[32] We refer here to the obvious adaptation to nets of the notion introduced in Definition 2.14 for sps.

PROOF. Let us define the *additive depth* of a link l of a net β as the number of $\&$-components of β containing l, and the additive depth of β as the maximal additive depth of its links.

The separation property (Lemma 5.4) implies that for every net β which contains n $\&$-links, if $\beta \xrightarrow{(\mathrm{ccad})\forall/\exists *} \beta'$, then the additive depth of β' is at most n. This allows to define a size on every $(\mathrm{ccad}), (\forall/\exists)$-reduct of β: consider the n-tuple whose i^{th} component is the number of links of the net having additive depth i, and order these n-tuples lexicographically. It is easy to show that this size shrinks at every (ccad) or $(\forall/\exists)$ step: suppose that $\beta \xrightarrow{\mathrm{ccad}\forall/\exists *} \beta' \xrightarrow{\mathtt{x}} \beta''$. If $\mathtt{x}$ is $(\forall/\exists)$, then at some (additive) depth $0 \leq i \leq n$ the number of links of β'' is strictly less than the number of links of depth i of β', and for every $0 \leq j \neq i \leq n$ the number of links of depth j in β' and β'' is the same. If $\mathtt{x}$ is (ccad), then let i be the depth of the $\&$-link l' of β' playing the role of the link l in the (ccad) step of Figure 19: at depth $j < i$ the number of links of β' and β'' is the same, at depth i there are strictly less links in β'' than in β', so that also in this case the previously defined size shrinks. □

We can conclude the subsection with the expected result:

Proposition 5.6. *Let β be a net. If $\mathrm{sl}(\beta) \in \mathrm{SN}$ then $\beta \in \mathrm{SN}$.*

PROOF. Immediate consequence of lemmata 5.3 and 5.5. □

5.4. Strong normalization for nets

Finally we prove that $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$, for every net β (Theorem 5.11). Here we need Girard's reducibility candidates. We give the particular definition of reducibility candidates (Definition 5.10) required to prove Theorem 5.11, and then we just sketch the proof of the theorem, which is standard after [7].

Definition 5.7. A *term of type* A is a net β together with a distinguished conclusion which is labelled by A. If β (resp. β') is a term of type A (resp. $A^\perp$), we denote by $\mathrm{CUT}(\beta, \beta')$ the net obtained by connecting β and β' by means of a cut with premises the two distinguished conclusions $A, A^\perp$.

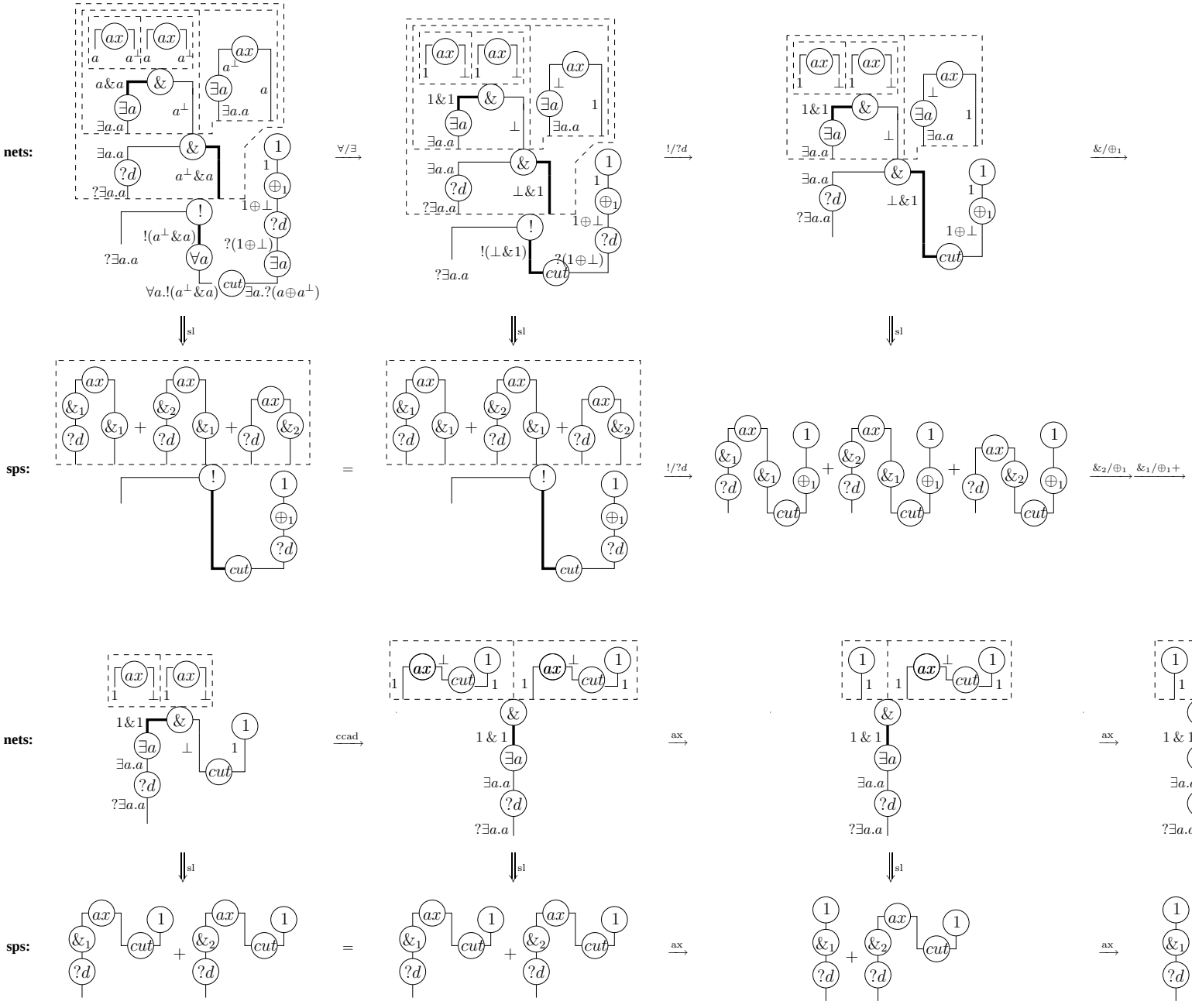
Definition 5.8 (duality). Let X be a set of terms of type A; we define $X^\perp$ as follows:

$$X^\perp = \left\{\beta' \text{ s.t. } \beta' \text{ term of type } A^\perp \text{ and } \mathrm{sl}(\mathrm{CUT}(\beta, \beta')) \in \mathrm{WN}^{\neg\mathsf{e}} \text{ for every } \beta \in X\right\}$$

Proposition 5.9. *Let X be a set of terms of type A:*

1. *if X contains the axiom link with conclusion $A, A^\perp$, then for every $\beta \in X^\perp$ one has $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$;*
2. *if for every $\beta \in X$ one has $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$, then $X^\perp$ contains the axiom link with conclusion $A, A^\perp$.*

PROOF. Immediate from the definitions. □


nets:
sps:
∀/∃
!/?d
&/⊕1
sl
=
ccad
ax
&2/⊕1
&1/⊕1+

Definition 5.10 (reducibility candidate). A *reducibility candidate of type* A is a set X of terms of type A s.t.:

1. $X \neq \emptyset$;
2. for every $\beta \in X$, one has $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg \mathsf{e}}$;
3. $X = X^{\perp\perp}$.

Theorem 5.11 ($\mathrm{WN}^{\neg \mathsf{e}}$ theorem). *If β is a net, then* $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg \mathsf{e}}$.

PROOF. Girard's proof of Theorem 4.26 ([7]) works perfectly if one substitutes "β is SN" with "$\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg \mathsf{e}}$". We just give here the idea of how to adapt the original proof. In what follows we use the notations and definitions of [7].

Let β be a net with conclusions $\mathbf{C}$ (where $\mathbf{C} = C_1, \ldots, C_n$ is a sequence of LL formulas), we prove that β is *reducible*, which means that the following holds (see Definition 4.26.8 of [7]): let $\mathbf{a}$ ($= a_1, \ldots, a_m$) be the list of all free variables of $\mathbf{C}$, then for every sequence of formulas $\mathbf{B}$ ($= B_1, \ldots, B_m$), every sequence $\mathbf{X}$ ($= X_1, \ldots, X_m$) of reducibility candidates of types $\mathbf{B}$ and every sequence of terms $\mathbf{t}$ ($= t_1, \ldots, t_n$) in $\mathrm{RED}(\mathbf{C}^{\perp}[\mathbf{X}/\mathbf{a}])$ (see Definition 4.26.6 of [7]), one has $\mathrm{sl}(\mathrm{CUT}(\beta[\mathbf{B}/\mathbf{a}]; \mathbf{t})) \in \mathrm{WN}^{\neg \mathsf{e}}$. The net $\mathrm{CUT}(\beta[\mathbf{B}/\mathbf{a}]; \mathbf{t})$ is obtained by:

- substituting in β the free variables $a_1, \ldots, a_m$ with the formulas $B_1, \ldots, B_m$: this yields $\beta[\mathbf{B}/\mathbf{a}]$
- given a term $t_i \in \mathrm{RED}({C_i}^{\perp}[\mathbf{X}/\mathbf{a}])$ for every conclusion $C_i[\mathbf{B}/\mathbf{a}]$ of $\beta[\mathbf{B}/\mathbf{a}]$, cutting all the t_i with $\beta[\mathbf{B}/\mathbf{a}]$.

The proof that $\mathrm{sl}(\mathrm{CUT}(\beta[\mathbf{B}/\mathbf{a}]; \mathbf{t})) \in \mathrm{WN}^{\neg \mathsf{e}}$ is by induction on a sequentialization of β, hence it splits in 16 cases (the number of rules of LL: ax, cut, $\otimes$, 1, $\parr$, $\perp$, $\&$, $\top$, $\oplus_1$, $\oplus_2$, !, $?w$, $?d$, $?c$, $\forall^2$, $\exists^2$). In every case, we have to prove that $\mathrm{sl}(\mathrm{CUT}(\beta[\mathbf{B}/\mathbf{a}]; \mathbf{t})) \in \mathrm{WN}^{\neg \mathsf{e}}$ whatever $t_i \in \mathrm{RED}({C_i}^{\perp}[\mathbf{X}/\mathbf{a}])$ has been selected. However, a simplification is often useful (and possible): for example in case ${C_i}^{\perp} = A \otimes B$ one needs to prove $\mathrm{sl}(\mathrm{CUT}(\beta[\mathbf{B}/\mathbf{a}]; \mathbf{t})) \in \mathrm{WN}^{\neg \mathsf{e}}$ only for those t_i belonging to $\mathrm{RED}({C_i}^{\perp}[\mathbf{X}/\mathbf{a}])$ and such that t_i can be obtained by performing a $\otimes$-link between some $t_i^1 \in \mathrm{RED}(A[\mathbf{X}/\mathbf{a}])$ and some $t_i^2 \in \mathrm{RED}(B[\mathbf{X}/\mathbf{a}])$. We shall use this simplification in the sequel of the proof. We consider only two cases: the $\&$- and !-cases, which show how works our version of reducibility candidates. Since the substitutions play no active role but make everything hard to read (and to write), we shall not indicate them (thus working with $\mathrm{RED}(\mathbf{C}^{\perp})$, etc...). We also use the fact that by definition of RED (see Definition 4.26.6 of [7]), one has $\mathrm{RED}(C^{\perp}) = \mathrm{RED}(C)^{\perp}$.

$\&$-case: β is obtained from β_1 and β_2 by the $\&$-box in Figure 18. After simplification, we see that we have to check that $\mathrm{sl}(\mathrm{CUT}(\beta; \mathbf{c}, \oplus_1 t)) \in \mathrm{WN}^{\neg \mathsf{e}}$ and $\mathrm{sl}(\mathrm{CUT}(\beta; \mathbf{c}, \oplus_2 u)) \in \mathrm{WN}^{\neg \mathsf{e}}$ for any $\mathbf{c} \in \mathrm{RED}(\mathbf{C})^{\perp}$, $t \in \mathrm{RED}(A)^{\perp}$, and $u \in \mathrm{RED}(B)^{\perp}$. By induction hypothesis $\mathrm{sl}(\mathrm{CUT}(\beta_1; \mathbf{c}, t)) \in \mathrm{WN}^{\neg \mathsf{e}}$ and $\mathrm{sl}(\mathrm{CUT}(\beta_2; \mathbf{c}, u)) \in \mathrm{WN}^{\neg \mathsf{e}}$ for any $\mathbf{c} \in \mathrm{RED}(\mathbf{C})^{\perp}$, $t \in \mathrm{RED}(A)^{\perp}$, and $u \in \mathrm{RED}(B)^{\perp}$. Since u can be chosen as an axiom, by Proposition 5.9 we deduce that $\mathrm{sl}(\mathrm{CUT}(\beta_2; \mathbf{c})) \in \mathrm{WN}^{\neg \mathsf{e}}$. Since by Definition 5.10 the slicing of every element of a reducibility candidate is $\mathrm{WN}^{\neg \mathsf{e}}$, we also have $\mathrm{sl}(\oplus_1 t) \in \mathrm{WN}^{\neg \mathsf{e}}$.

Consider now $\mathrm{sl}(\mathrm{CUT}(\beta;\mathbf{c},\oplus_1 t)) = \sigma = \sigma_1 + \sigma_2$, where σ_1 (resp. σ_2) is the multiset of the slices of $\mathrm{CUT}(\beta;\mathbf{c},\oplus_1 t)$ which contain the component β_1 (resp. β_2) of the &-box. Notice that $\sigma_1 \xrightarrow{\&_1/\oplus_1} \mathrm{sl}(\mathrm{CUT}(\beta_1;\mathbf{c},t))$, hence $\sigma_1 \in \mathrm{WN}^{\neg\mathsf{e}}$. But pay attention that we cannot yet infer that also $\sigma \in \mathrm{WN}^{\neg\mathsf{e}}$, because for reducing σ to $\mathrm{sl}(\mathrm{CUT}(\beta_1;\mathbf{c},t))$ we should perform several $(\&_2/\oplus_1)$ reductions, which are erasing. Nevertheless we deduce that σ_2 is $\mathrm{WN}^{\neg\mathsf{e}}$ since $\mathrm{sl}(\mathrm{CUT}(\beta_2;\mathbf{c})) \in \mathrm{WN}^{\neg\mathsf{e}}$ and $\mathrm{sl}(\oplus_1 t) \in \mathrm{WN}^{\neg\mathsf{e}}$ and σ_2 is obtained by connecting every slice of $\mathrm{sl}(\mathrm{CUT}(\beta_2;\mathbf{c}))$ and every slice of $\mathrm{sl}(\oplus_1 t)$ by means of a $(\&_2/\oplus_1)$ cut link. We conclude that $\mathrm{sl}(\mathrm{CUT}(\beta;\mathbf{c},\oplus_1 t)) \in \mathrm{WN}^{\neg\mathsf{e}}$.

For symmetric reasons, $\mathrm{sl}(\mathrm{CUT}(\beta;\mathbf{c},\oplus_2 u)) \in \mathrm{WN}^{\neg\mathsf{e}}$.

!-case: β is a !-link with conclusions $!A, ?C_1, \ldots, ?C_n$ whose associated box is the net β'. The induction hypothesis is that $\mathrm{sl}(\mathrm{CUT}(\beta';\mathbf{c},t)) \in \mathrm{WN}^{\neg\mathsf{e}}$ for all $\mathbf{c} \in \mathrm{RED}(?\mathbf{C})^\perp$ and $t \in \mathrm{RED}(A)^\perp$; and we want to conclude that $\mathrm{sl}(\mathrm{CUT}(\beta;\mathbf{c},u)) \in \mathrm{WN}^{\neg\mathsf{e}}$ for all $\mathbf{c} \in \mathrm{RED}(?\mathbf{C})^\perp$ and for all u in $\mathrm{RED}(!A)^\perp$. Now, it is easy to show that we can make a simplification on the whole sequence $\mathbf{c}$, namely that $\mathbf{c}$ is of the form $!\mathbf{d}$, for $\mathbf{d} \in \mathrm{RED}(\mathbf{C})^\perp$. By several (!/!) reduction steps we reduce $\mathrm{CUT}(\beta;\mathbf{c},u)$ to $\mathrm{CUT}(!\,\mathrm{CUT}(\beta';\mathbf{c}),u)$, where $!\,\mathrm{CUT}(\beta';\mathbf{c})$ is the net consisting in a !-link with conclusion $!A$, whose associated box is the net $\mathrm{CUT}(\beta';\mathbf{c})$ with conclusions A. Now, by induction hypothesis $\mathrm{CUT}(\beta';\mathbf{c}) \in \mathrm{RED}(A)$, hence $!\,\mathrm{CUT}(\beta';\mathbf{c}) \in !\,\mathrm{RED}(A) \subset \mathrm{RED}(!A)$. So $\mathrm{sl}(\mathrm{CUT}(!\,\mathrm{CUT}(\beta';\mathbf{c}),u)) \in \mathrm{WN}^{\neg\mathsf{e}}$. Finally, since $\mathrm{sl}(\mathrm{CUT}(\beta;\mathbf{c},u)) \xrightarrow{!/!*} \mathrm{sl}(\mathrm{CUT}(!\,\mathrm{CUT}(\beta';\mathbf{c}),u))$, we conclude that $\mathrm{sl}(\mathrm{CUT}(\beta;\mathbf{c},u)) \in \mathrm{WN}^{\neg\mathsf{e}}$.

□

We can eventually conclude:

Theorem 5.12 (SN **theorem).** *If β is a net, then $\beta \in \mathrm{SN}$.*

PROOF. If β is a net, then $\mathrm{sl}(\beta)$ satisfies AC (Proposition 5.1) and $\mathrm{sl}(\beta) \in \mathrm{WN}^{\neg\mathsf{e}}$ (Theorem 5.11). We can then apply Theorem 4.2 and prove that $\mathrm{sl}(\beta) \in \mathrm{SN}$, from which we conclude that $\beta \in \mathrm{SN}$ (Proposition 5.6). □

Aknowledgements

We thank Jean-Yves Girard for his course on proof-theory given in Roma Tre in fall 2004, where he presented Gandy's method for natural deduction. We also thank Paolo Tranquilli, Olivier Laurent and Michele Abrusci for several discussions on the subject.

References

[1] Barendregt, H., 1984. The lambda calculus, its syntax and semantics, 2nd Edition. No. 103 in Studies in Logic and the Foundations of Mathematics. North-Holland.

[2] Danos, V., 1990. La logique linéaire appliquée à l'étude de divers processus de normalisation (principalement du λ-calcul). Thèse de doctorat, Université Paris VII.

[3] Danos, V., Joinet, J.-B., Schellinx, H., Sep. 1997. A new deconstructive logic: linear logic. Journal of Symbolic Logic 62 (3), 755–807.

[4] Di Cosmo, R., Kesner, D., Polonovski, E., Jun. 2003. Proof nets and explicit substitutions. Mathematical Structures in Computer Science 13 (3), 409–450.

[5] Gandy, R. O., 1980. Proofs of strong normalization. In: Seldin, J., Hindley, J. (Eds.), To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism. Academic Press Limited, pp. 457–477.

[6] Girard, J.-Y., 1972. Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur. Thèse de doctorat d'etat, Université Paris VII.

[7] Girard, J.-Y., 1987. Linear logic. Theoretical Computer Science 50, 1–102.

[8] Girard, J.-Y., 1987. Proof Theory and Logical Complexity. Studies in Proof-theory. Bibliopolis, Napoli.

[9] Girard, J.-Y., 1991. Quantifiers in linear logic II. In: Corsi, Sambin (Eds.), Nuovi problemi della logica e della filosofia della scienza. CLUEB, Bologna, pp. 79–90.

[10] Girard, J.-Y., 1996. Proof-nets: the parallel syntax for proof-theory. In: Ursini, A., Agliano, P. (Eds.), Logic and Algebra. Vol. 180 of Lecture Notes In Pure and Applied Mathematics. Marcel Dekker, New York, pp. 97–124.

[11] Hughes, D., van Glabbeek, R., Jun. 2003. Proof nets for unit-free multiplicative-additive linear logic. In: Proceedings of the eighteenth annual symposium on Logic in Computer Science. IEEE, IEEE Comp. Soc. Press, Ottawa, pp. 1–10.

[12] Klop, J. W., Jun. 1992. Term rewriting systems. In: Abramsky, S., Gabbay, D., Maibaum, T. (Eds.), Handbook of Logic in Computer Science. Vol. 2. IEEE, Oxford University Press, pp. 1–116.

[13] Krivine, J.-L., 1990. Lambda-Calcul : Types et Modèles. Études et Recherches en Informatique. Masson.

[14] Laurent, O., Mar. 2002. Étude de la polarisation en logique. Thèse de doctorat, Université Aix-Marseille II.

[15] Laurent, O., Quatrini, M., Tortora de Falco, L., Jul. 2005. Polarized and focalized linear and classical proofs. Annals of Pure and Applied Logic 134 (2–3), 217–264.

[16] Laurent, O., Tortora de Falco, L., Nov. 2004. Slicing polarized additive normalization. In: Ehrhard, T., Girard, J.-Y., Ruet, P., Scott, P. (Eds.), Linear Logic in Computer Science. Vol. 316 of London Mathematical Society Lecture Note Series. Cambridge University Press, pp. 247–282.

[17] Laurent, O., Tortora de Falco, L., 2006. Obsessional cliques: a semantic characterization of bounded time complexity. In: Proceedings of the twenty-first annual IEEE symposium on Logic In Computer Science (LICS '06).

[18] Okada, M., 1999. Phase semantic cut-elimination and normalization proofs of first- and higher-order linear logic. Theor. Comput. Sci. 227 (1-2), 333–396.

[19] Prawitz, D., 1965. Natural Deduction. A Proof.Theoretical Study. Almquist and Wiksell, Stockholm, Sweden.

[20] Regnier, L., 1992. Lambda-calcul et réseaux. Thèse de doctorat, Université Paris VII.

[21] Sørensen, M. H., 1997. SN from WN in typed λ-calculi. Information and Computation 133 (1), 35–71.

[22] Terese, 2003. Term Rewriting Systems. Vol. 55 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

[23] Tortora de Falco, L., Jan. 2000. Réseaux, cohérence et expériences obsessionnelles. Thèse de doctorat, Université Paris VII.

[24] Tortora de Falco, L., 2003. Additives of linear logic and normalization- part I: a (restricted) church-rosser property. Theoretical Computer Science 294/3, 489–524.

[25] Tranquilli, P., 2008. Intuitionistic differential nets and lambda calculus, to appear in: *Girard's Festschrift, special issue Theoretical Computer Science*.

[26] Tranquilli, P., 2009. Confluence of pure differential nets with promotion, to appear in: *Proceedings of Computer Science Logic 2009*.