June 16, 2005 — Final Version for "Structures and Deduction 2005", Lisbon

From Deep Inference to Proof Nets

Lutz Straßburger

Universität des Saarlandes — Informatik — Programmiersysteme
Postfach 15 11 50 — 66041 Saarbrücken — Germany
`http://www.ps.uni-sb.de/~lutz`

Abstract. This paper shows how derivations in (a variation of) SKS can be translated into proof nets. Since an SKS derivation contains more information about a proof than the corresponding proof net, we observe a loss of information which can be understood as "eliminating bureaucracy". Technically this is achieved by cut reduction on proof nets. As an intermediate step between the two extremes, SKS derivations and proof nets, we will see nets representing derivations in "Formalism A".

1 Introduction

Through the development of the two concepts of *deep inference* [Gug02] and *proof nets* [Gir87] the quest for the identity of proofs has become fashionable again, and the research on the fundamental question "When are two proofs the same?" seems now to be booming.

Proof nets have been conceived by Girard [Gir87] in order to avoid *bureaucracy*: in formal systems like the sequent calculus two proofs that are "morally the same" are distinguished by trivial rule permutations.

Deep inference has been conceived by Guglielmi in order to obtain a deductive system for a non-commutative logic [Gug02]. In a formalism employing deep inference, like the calculus of structures, one can apply inference rules anywhere deep inside formulae as we know it from term rewriting, instead of decomposing formulae along their main connectives as we know it from traditional formalisms. From the "we-wish-to-eliminate-bureaucracy" point of view, this is a disaster: The number of possible "trivial rule permutations" explodes, compared to the sequent calculus. However, the finer granularity of inference rules (one inference step in the sequent calculus corresponds to many inference steps in the calculus of structures) allows a finer analysis of the inner structure of proofs, which in turn can lead to new notions of proof nets (as happened in [SL04] and [LS05b]).

In this paper we will see how proof nets can be extracted directly from deep inference systems. I will concentrate here only on classical logic, more precisely on (a slight variation of) system SKS [BT01,Brü03a], the most popular system for classical logic in the calculus of structures. But it should be clear that the exercise of this paper can in the same way be carried out for any other system, in particular also for linear logic as it is presented in [Str02].

To some extend, one can say that proof nets make as many identifications between proofs as possible (without ending up in a triviality), and derivations in the calculus of structures makes as few identifications as possible. These two extremes span a whole universe of possible proof identifications. And going from the extreme with few identifications to the extreme with many identification means losing information, namely, the "bureaucratic" information that makes the additional distinctions. I will argue, that this process of losing information can be modelled by cut elimination. In each single cut reduction step some bit of information is lost. Depending on the restrictions on cut elimination one can choose which information to lose.

The question, when this information is bureaucratic and when it is non-bureaucratic (i.e., essential for the proof), must be left unanswered in this paper.

2 Proof Nets for Classical Logic

Proof nets are abstract (graphical) presentations of proofs such that all "trivial rule permutations" are quotiented away. Ideally the notion of proof net should be independent from any syntactic formalism. But due to the almost absolute monopoly of the sequent calculus, most notions of proof nets proposed in the past related themselves to the sequent calculus. Consequently we could observe features like "boxes" and explicit "contraction links". The latter appeared not only in linear logic [Gir96] but also in classical logic (as sketched in [Gir91] and detailed out in [Rob03]). The slogan of the early proof nets was

Slogan 1: *Every link in the proof net corresponds to a rule application in the sequent calculus.*

with the basic idea that if two rules "trivially permute" in the sequent calculus, then the corresponding links in the proof net are independent. However, more recent proposals for proof nets follow a different slogan:

Slogan 2: *A proof net is a formula tree (or sequent forest) enriched with additional graph structure.*

This additional graph structure is supposed to capture the *essence* of the proof. To our knowledge the first notion of proof net in this more modern setting were [HvG03] for unit-free multiplicative additive linear logic (MALL) and [SL04] for multiplicative linear logic (MLL) with units.[1] Then in [LS05b] proof nets for classical logic obeying Slogan 2 followed. Let me now recall that latter notion of proof nets. (I consider here only the $\mathbb{N}$-nets of [LS05b].)

The set of *formulae* is generated via the binary connectives $\wedge$ (*conjunction*) and $\vee$ (*disjunction*) from the set $\mathcal{A} \cup \bar{\mathcal{A}} \cup \{\mathbf{t}, \mathbf{f}\}$, where $\mathcal{A} = \{a, b, c, \ldots\}$ is a countable set of *propositional variables* and $\bar{\mathcal{A}} = \{\bar{a}, \bar{b}, \bar{c}, \ldots\}$ is the set of *negated propositional variables*, and $\mathbf{t}$ and $\mathbf{f}$ are the *constants* representing "true" and

[1] In fact, the first has been [Gir87] (or more precisely [KM71]) simply because for the special case of unit-free MLL both slogans coincide: every connective in the formulae corresponds to an application of a sequent rule, and the axiom links attached to the formulae capture exactly the essence of a proof in unit-free MLL. This very fortunate coincidence is also the reason why proof nets for unit-free MLL behave so remarkably well and were so successful from the very beginning.

"false", respectively. The elements of the set $\mathcal{A} \cup \bar{\mathcal{A}} \cup \{\mathbf{t}, \mathbf{f}\}$ are called *atoms*. A finite list $\Gamma = A_1, A_2, \ldots, A_n$ of formulae is called a *sequent*. I will consider formulae as binary trees (and sequents as forests), whose leaves are decorated by atoms, and whose inner nodes are decorated by the connectives. The negation $\bar{A}$ of a formula A is defined as follows:

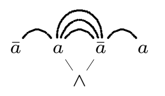
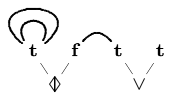
$$\bar{\bar{a}} = a \qquad \bar{\mathbf{t}} = \mathbf{f} \qquad \bar{\mathbf{f}} = \mathbf{t} \qquad \overline{(A \wedge B)} = \bar{B} \vee \bar{A} \qquad \overline{(A \vee B)} = \bar{B} \wedge \bar{A} \tag{1}$$

Here a ranges over the set $\mathcal{A}$. However, from now on I will use a to denote an arbitrary atom (including constants). Note that $\bar{\bar{A}} = A$ for all A.

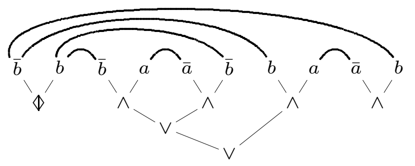
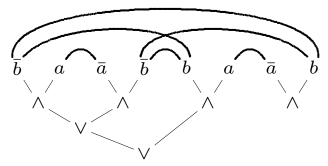
There is a special kind of auxiliary formula, called *cut*, which is of the shape $B \mathbin{\Diamond} \bar{B}$, where $\Diamond$ is called the *cut connective* and is allowed only at the root of a formula tree. A *cut sequent* is a finite list $\Sigma = B_1 \mathbin{\Diamond} \bar{B}_1, \ldots, B_n \mathbin{\Diamond} \bar{B}_n$ of cuts.

A *prenet* $P, \Sigma \triangleright \Gamma$ consists of a sequent Γ, a cut sequent Σ, and an undirected multi-graph P whose set of vertices is the set of leaves of Γ and Σ and whose set of edges obeys the following conditions: (i) whenever there is an edge between two leaves, then one is decorated by an atom a and the other by its dual $\bar{a}$, and (ii) whenever there is an edge connecting a leaf to itself, then this leaf is decorated by $\mathbf{t}$.

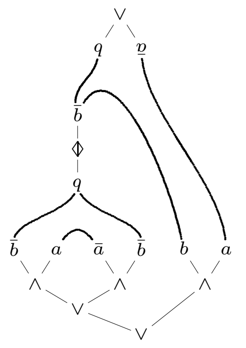
One can think of P also of an undirected graph whose edges are labeled by natural numbers (hence the name $\mathbb{N}$-net in [LS05b]), but here I will draw it as multi-graph, for example:

 
$\bar{a}$ a $\bar{a}$ a $\wedge$
$\mathbf{t}$ $\mathbf{f}$ $\mathbf{t}$ $\mathbf{t}$ $\Diamond$ $\vee$

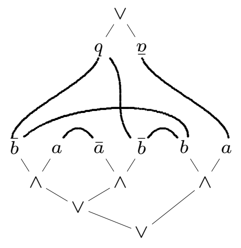
In the following, I will consider only nets where Γ contains exactly two formulae (there is no restriction on the number of cuts in Σ). Here are two examples, one with and one without cuts (both are variations of examples in [LS05b]):

These can also be drawn as



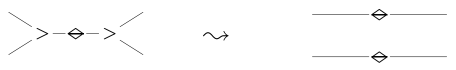
and



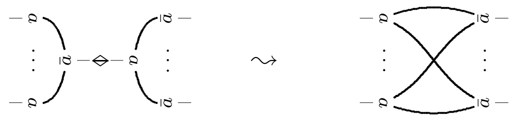
(2)

which will be the preferred way from now on.

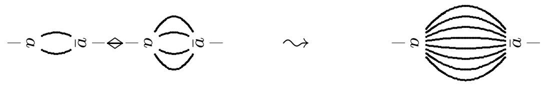
The cut reduction procedure for prenets is defined as follows. For cuts on compound formulae we have:

 (3)

(For saving space, the picture is put on the side.) Atomic cuts are reduced as follows:

 (4)

This means that for every pair of edges, where one is connected to the cut on one side and the other is connected to the other side of the cut, there is in the reduced net an edge connecting the two other ends of the pair (an edge connecting the two ends of the cut disappears with the cut). For the formal details, see [LS05b]. Here, let me only draw attention to the fact that if there is more than one edge between two dual atoms, the number of edges is multiplied, as in the following example:

 (5)

This causes an exponential increase of the number of edges (in the number of atomic cuts) during the cut reduction process.

Obviously the cut reduction on prenets is terminating. But we have confluency only in the case where the number of edges between two atoms is restricted to one, i.e, where the multi-graph is just a graph (as shown in [LS05b]). In that case also the correctness criterion of [LS05b] is adequate. However, at the current state of the art, there is no suitable criterion yet for the general case. For that reason the objects here are called "prenets". The term "proof net" should be reserved to those objects that actually represent proofs.

3 Deep Inference for Classical Logic

Deep inference is a new paradigm for proof theoretical formalisms. The most prominent example of a formalism employing deep inference is the calculus of structures. It has successfully been employed to give new presentations for many logics, including classical logic [BT01,Brü03a], minimal logic [Brü03b], intuitionistic logic [Tiu05], several modal logics [SS03,Sto04], linear logic [Str03], and various noncommutative logics [DG04,Gug02,GS02].

Let me now recall the deep inference sytem SKS for classical logic [BT01]. At the same time I modify it slightly: I remove the syntactical equivalence employed by the calculus of structures and represent all the defining equations by inference rules. Nonetheless, I use the "outfix" notation employed by the calculus of structures because derivations are much easier to read that way. That

$$\mathsf{ai}\!\downarrow \frac{S\{\mathbf{t}\}}{S[a,\bar{a}]} \qquad \mathsf{ai}\!\uparrow \frac{S(a,\bar{a})}{S\{\mathbf{f}\}}$$

$$\mathsf{s}\,\frac{S(A,[B,C])}{S[(A,B),C]}$$

$$\mathsf{aw}\!\downarrow \frac{S\{\mathbf{f}\}}{S\{a\}} \qquad \mathsf{ac}\!\downarrow \frac{S[a,a]}{S\{a\}} \qquad \mathsf{ac}\!\uparrow \frac{S\{a\}}{S(a,a)} \qquad \mathsf{aw}\!\uparrow \frac{S\{a\}}{S\{\mathbf{t}\}}$$

$$\mathsf{nm}\!\downarrow \frac{S\{\mathbf{f}\}}{S(\mathbf{f},\mathbf{f})} \qquad \mathsf{m}\,\frac{S[(A,C),(B,D)]}{S([A,B],[C,D])} \qquad \mathsf{nm}\!\uparrow \frac{S[\mathbf{t},\mathbf{t}]}{S\{\mathbf{t}\}}$$

$$\sigma\!\downarrow \frac{S[A,B]}{S[B,A]} \qquad \alpha\!\downarrow \frac{S[A,[B,C]]}{S[[A,B],C]} \qquad \alpha\!\uparrow \frac{S(A,(B,C))}{S((A,B),C)} \qquad \sigma\!\uparrow \frac{S(A,B)}{S(B,A)}$$

$$\mathbf{f}\!\downarrow \frac{S\{A\}}{S[A,\mathbf{f}]} \qquad \mathbf{t}\!\downarrow \frac{S\{A\}}{S(A,\mathbf{t})} \qquad \mathbf{t}\!\uparrow \frac{S[\mathbf{f},A]}{S\{A\}} \qquad \mathbf{f}\!\uparrow \frac{S(\mathbf{t},A)}{S\{A\}}$$

Fig. 1. The inference rules of system SKS

means I write $[A,B]$ for $A \vee B$ and (A,B) for $A \wedge B$. For example the formula $((\bar{b} \wedge a) \vee (\bar{a} \wedge \bar{b})) \vee (b \wedge a)$ is in this notation written as $[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]$.

Before presenting the rules of the system we need to introduce the notion of a context, which is simply a formula with a hole. It is usually denoted by $S\{\ \}$. For example $[[(\bar{b},a),\{\ \}],(b,a)]$ is a context. Let it be denoted by $S\{\ \}$, and let $A = (\bar{a},\bar{b})$. Then $S\{A\} = [[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]$. I will omit the context-braces when structural parentheses fill the hole exactly. For example $S(\bar{a},\bar{b})$ stands for $S\{(\bar{a},\bar{b})\}$.

Now we are ready to see the inference rules. In this paper, they are all of the shape

$$\rho\,\frac{S\{A\}}{S\{B\}}$$

and this simply specifies a step of rewriting (via the implication $A \Rightarrow B$) inside a generic context $S\{\ \}$. Such a rule is called *deep*, which is the reason for the term "deep inference". If a rule scheme does not have this generic context (or there are size restrictions to the context), then the rule is called *shallow*.

The inference rules of *system* SKS (which are all deep) are shown in Figure 1. The rules ai↓ and ai↑ are called *atomic identity* and *atomic cut* (the i stands for "interaction").[2] The rules s and m are called *switch* and *medial*, respectively.

[2] Here we can make an important observation: There are two very different notions of "cut" and the two should not be mixed up. On the one side we have the cut as a rule, and cut elimination means that this rule is admissible. In the calculus of structures it means that the whole up-fragment of the system (i.e., all rules with the ↑ in the name) are admissible. This holds in particular also for system SKS, see [Brü03a],

They are the soul of system SKS. Note that these two rules are self-dual, while all other rules have their dual "co-rule". For example, the rules $\mathsf{aw}\downarrow$ and $\mathsf{ac}\downarrow$ (called *atomic weakening down* and *atomic contraction down*, respectively) have as duals the rules $\mathsf{aw}\uparrow$ and $\mathsf{ac}\uparrow$, which are called *atomic weakening up* and *atomic contraction up*, respectively.[3] The rules $\mathsf{nm}\downarrow$ and $\mathsf{nm}\uparrow$ are called *nullary medial* (*up* and *down*). At this point it might seem rather strange that they are in the system. After all, they are instances of the atomic contraction rules. There are two reasons: The first one is that in a system in the calculus of structures the complete up-fragment should be admissible (i.e., also the rule $\mathsf{ac}\uparrow$), but we need $\mathsf{nm}\downarrow$ for completeness. The second reason is that the two rules $\mathsf{ac}\uparrow$ and $\mathsf{nm}\downarrow$ look similar only from the outside. When we look at the inside of the rules—we will do this in Section 5—we can see that they are of a very different nature.

The other rules ($\alpha\downarrow$, $\alpha\uparrow$, $\sigma\downarrow$, $\sigma\uparrow$, $\mathbf{t}\downarrow$, $\mathbf{t}\uparrow$, $\mathbf{f}\downarrow$, $\mathbf{f}\uparrow$) just say that $\wedge$ and $\vee$ are associative and commutative and that $\mathbf{t}$ and $\mathbf{f}$ are the units for them. Usually, in systems in the calculus of structures, formulae are considered up to an syntactic equivalence incorporating the associativity and commutativity (and the units) of the binary connectives. For example $[[A, B], [C, \mathbf{f}]]$ and $[B, [(\mathbf{t}, A), C]]$ are considered the same and denoted by $[A, B, C]$. Here, I deviate from this practice because having explicit rules for associativity and commutativity simplifies the translation to derivation nets in Section 5.

But before that, we need to plug our rules together to get *derivations*, which are finite chains of instances of inference rules. The topmost formula in a derivation is called its *premise* and the bottommost formula is called the *conclusion.* A derivation Δ, whose premise is A, whose conclusion is B, and whose inference rules are all contained in the system S, is denoted by

$$\begin{array}{c} A \\ \Delta \big\Vert \mathsf{S} \\ B \end{array} \quad .$$

Figure 2 shows two examples of derivations in system SKS.

4 The ABC of Bureaucracy

The term "bureaucracy" is used to describe the phenomenon that oftentimes two formal proofs in a certain formalism denote "morally" the same proof but differ due to trivial rule permutations or other syntactic phenomena. Of course, the main problem here is to decide when two proofs should be "morally" the same. I.e., when is a certain syntactic phenomenon an important information about the proof and when is it just "bureaucracy"?

but is not of relevance for this paper. On the other side we have the cut $\Diamond$, and cut elimination means composition of derivations (or proofs, or arrows in a category), and this will play a role in this paper.

[3] In system SKS the rules $\mathsf{ai}\downarrow$, $\mathsf{ac}\downarrow$, and $\mathsf{aw}\downarrow$ are atomic because their general counterparts $\mathsf{i}\downarrow$, $\mathsf{c}\downarrow$, and $\mathsf{w}\downarrow$ are derivable. Dually for the up-rules (see [Brü03a] for details).

$$
\begin{array}{rc}
 & [\bar{b}, a] \\ \cline{2-2}
\mathsf{t}{\downarrow} & [\bar{b}, (a, \mathbf{t})] \\ \cline{2-2}
\mathsf{ai}{\downarrow} & [\bar{b}, (a, [b, \bar{b}])] \\ \cline{2-2}
\mathsf{s} & [\bar{b}, [(a, b), \bar{b}]] \\ \cline{2-2}
\sigma{\uparrow} & [\bar{b}, [(b, a), \bar{b}]] \\ \cline{2-2}
\sigma{\downarrow} & [\bar{b}, [\bar{b}, (b, a)]] \\ \cline{2-2}
\alpha{\downarrow} & [[\bar{b}, \bar{b}], (b, a)] \\ \cline{2-2}
\mathsf{ac}{\downarrow} & [\bar{b}, (b, a)] \\ \cline{2-2}
\mathsf{ac}{\uparrow} & [(\bar{b}, \bar{b}), (b, a)] \\ \cline{2-2}
\mathsf{t}{\downarrow} & [(\bar{b}, (\bar{b}, \mathbf{t})), (b, a)] \\ \cline{2-2}
\mathsf{ai}{\downarrow} & [(\bar{b}, (\bar{b}, [a, \bar{a}])), (b, a)] \\ \cline{2-2}
\mathsf{s} & [(\bar{b}, [(\bar{b}, \bar{a}), a]), (b, a)] \\ \cline{2-2}
\sigma{\uparrow} & [(\bar{b}, [(\bar{a}, \bar{b}), a]), (b, a)] \\ \cline{2-2}
\sigma{\downarrow} & [(\bar{b}, [a, (\bar{a}, \bar{b})]), (b, a)] \\ \cline{2-2}
\mathsf{s} & [[(\bar{b}, a), (\bar{a}, \bar{b})], (b, a)]
\end{array}
\qquad \text{and} \qquad
\begin{array}{rc}
 & [\bar{b}, a] \\ \cline{2-2}
\mathsf{t}{\downarrow} & [\bar{b}, (a, \mathbf{t})] \\ \cline{2-2}
\mathsf{ai}{\downarrow} & [\bar{b}, (a, [b, \bar{b}])] \\ \cline{2-2}
\mathsf{t}{\downarrow} & [\bar{b}, (a, [b, (\bar{b}, \mathbf{t})])] \\ \cline{2-2}
\mathsf{ai}{\downarrow} & [\bar{b}, (a, [b, (\bar{b}, [\bar{b}, b])])] \\ \cline{2-2}
\mathsf{s} & [\bar{b}, (a, [b, [(\bar{b}, \bar{b}), b]])] \\ \cline{2-2}
\sigma{\downarrow} & [\bar{b}, (a, [b, [b, (\bar{b}, \bar{b})]])] \\ \cline{2-2}
\alpha{\downarrow} & [\bar{b}, (a, [[b, b], (\bar{b}, \bar{b})])] \\ \cline{2-2}
\mathsf{s} & [\bar{b}, [(a, [b, b]), (\bar{b}, \bar{b})]] \\ \cline{2-2}
\sigma{\downarrow} & [\bar{b}, [(\bar{b}, \bar{b}), (a, [b, b])]] \\ \cline{2-2}
\sigma{\uparrow} & [\bar{b}, [(\bar{b}, \bar{b}), ([b, b], a)]] \\ \cline{2-2}
\alpha{\downarrow} & [[\bar{b}, (\bar{b}, \bar{b})], ([b, b], a)] \\ \cline{2-2}
\mathsf{ac}{\uparrow} & [[(\bar{b}, \bar{b}), (\bar{b}, \bar{b})], ([b, b], a)] \\ \cline{2-2}
\mathsf{ac}{\downarrow} & [[(\bar{b}, \bar{b}), (\bar{b}, \bar{b})], (b, a)] \\ \cline{2-2}
\mathsf{m} & [([\bar{b}, \bar{b}], [\bar{b}, \bar{b}]), (b, a)] \\ \cline{2-2}
\mathsf{ac}{\downarrow} & [(\bar{b}, [\bar{b}, \bar{b}]), (b, a)] \\ \cline{2-2}
\mathsf{ac}{\downarrow} & [(\bar{b}, \bar{b}), (b, a)] \\ \cline{2-2}
\mathsf{t}{\downarrow} & [(\bar{b}, (\bar{b}, \mathbf{t})), (b, a)] \\ \cline{2-2}
\mathsf{ai}{\downarrow} & [(\bar{b}, (\bar{b}, [a, \bar{a}])), (b, a)] \\ \cline{2-2}
\mathsf{s} & [(\bar{b}, [(\bar{b}, \bar{a}), a]), (b, a)] \\ \cline{2-2}
\sigma{\uparrow} & [(\bar{b}, [(\bar{a}, \bar{b}), a]), (b, a)] \\ \cline{2-2}
\sigma{\downarrow} & [(\bar{b}, [a, (\bar{a}, \bar{b})]), (b, a)] \\ \cline{2-2}
\mathsf{s} & [[(\bar{b}, a), (\bar{a}, \bar{b})], (b, a)]
\end{array}
$$

Fig. 2. Two examples of derivations

Consider now the right derivation in Figure 2. It is intuitively clear that we would not change the essence of the derivation if we exchanded the two $\mathsf{ac}{\downarrow}$ between the m and the $\mathbf{t}{\downarrow}$ in the lower half of the derivation. That the two $\mathsf{ac}{\downarrow}$ are ordered one above the other can be considered to be an act of "bureaucracy". In fact, following this intuition, we can permute the first $\mathsf{ac}{\downarrow}$ almost all the way down in the derivation, as shown in Figure 3. This kind of "bureaucracy" has been dubbed *bureaucracy of type A* by Guglielmi [Gug04a]. More generally whenever there is a derivation Δ from A to B and a derivation Δ' from C to D, then

$$
\begin{array}{c}
(A, C) \\
{\scriptstyle \Delta}\big\Vert \\
(B, C) \\
{\scriptstyle \Delta'}\big\Vert \\
(B, D)
\end{array}
\qquad \text{and} \qquad
\begin{array}{c}
(A, C) \\
{\scriptstyle \Delta'}\big\Vert \\
(A, D) \\
{\scriptstyle \Delta}\big\Vert \\
(B, D)
\end{array}
\tag{6}
$$

as well as any other "merge" of Δ and Δ' should be considered to be the same. Guglielmi calls a formalism which *per se* makes these identifications *Formalism A*. However, Formalism A does not allow the identification of the two derivations in Figure 4. where the $\mathsf{ac}{\downarrow}$ is not "next to" another derivation but "inside" another derivation. This phenomenon is called *bureaucracy of type B* [Gug04b]. Then *Formalism B* is a formalism that avoids this kind of bureaucracy.

Besides bureaucracy of type A and B, we can observe another kind of bureaucracy, which we will call here *bureaucracy of type C*. Consider the two derivations in Figure 5. They can considered to be essentially the same, because in both the

$$
\begin{array}{rc}
 & [([\bar{b},\bar{b}],[\bar{b},\bar{b}]),(b,a)] \\
\mathsf{ac}\!\downarrow & \overline{[(\bar{b},[\bar{b},\bar{b}]),(b,a)]} \\
\mathsf{ac}\!\downarrow & \overline{[(\bar{b},\bar{b}),(b,a)]} \\
\mathbf{t}\!\downarrow & \overline{[(\bar{b},(\bar{b},\mathbf{t})),(b,a)]} \\
\mathsf{ai}\!\downarrow & \overline{[(\bar{b},(\bar{b},[a,\bar{a}])),(b,a)]} \\
\mathsf{s} & \overline{[(\bar{b},[(\bar{b},\bar{a}),a]),(b,a)]} \\
\sigma\!\uparrow & \overline{[(\bar{b},[(\bar{a},\bar{b}),a]),(b,a)]} \\
\sigma\!\downarrow & \overline{[(\bar{b},[a,(\bar{a},\bar{b})]),(b,a)]} \\
\mathsf{s} & \overline{[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]}
\end{array}
\quad \rightsquigarrow \quad
\begin{array}{rc}
 & [([\bar{b},\bar{b}],[\bar{b},\bar{b}]),(b,a)] \\
\mathsf{ac}\!\downarrow & \overline{[([\bar{b},\bar{b}],\bar{b}),(b,a)]} \\
\mathbf{t}\!\downarrow & \overline{[([\bar{b},\bar{b}],(\bar{b},\mathbf{t})),(b,a)]} \\
\mathsf{ai}\!\downarrow & \overline{[([\bar{b},\bar{b}],(\bar{b},[a,\bar{a}])),(b,a)]} \\
\mathsf{s} & \overline{[([\bar{b},\bar{b}],[(\bar{b},\bar{a}),a]),(b,a)]} \\
\sigma\!\uparrow & \overline{[([\bar{b},\bar{b}],[(\bar{a},\bar{b}),a]),(b,a)]} \\
\sigma\!\downarrow & \overline{[([\bar{b},\bar{b}],[a,(\bar{a},\bar{b})]),(b,a)]} \\
\mathsf{ac}\!\downarrow & \overline{[(\bar{b},[a,(\bar{a},\bar{b})]),(b,a)]} \\
\mathsf{s} & \overline{[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]}
\end{array}
$$

Fig. 3. Example for type-A bureaucracy

$$
\begin{array}{rc}
 & [([\bar{b},\bar{b}],[\bar{b},\bar{b}]),(b,a)] \\
\mathsf{ac}\!\downarrow & \overline{[([\bar{b},\bar{b}],\bar{b}),(b,a)]} \\
\mathbf{t}\!\downarrow & \overline{[([\bar{b},\bar{b}],(\bar{b},\mathbf{t})),(b,a)]} \\
\mathsf{ai}\!\downarrow & \overline{[([\bar{b},\bar{b}],(\bar{b},[a,\bar{a}])),(b,a)]} \\
\mathsf{s} & \overline{[([\bar{b},\bar{b}],[(\bar{b},\bar{a}),a]),(b,a)]} \\
\sigma\!\uparrow & \overline{[([\bar{b},\bar{b}],[(\bar{a},\bar{b}),a]),(b,a)]} \\
\sigma\!\downarrow & \overline{[([\bar{b},\bar{b}],[a,(\bar{a},\bar{b})]),(b,a)]} \\
\mathsf{ac}\!\downarrow & \overline{[(\bar{b},[a,(\bar{a},\bar{b})]),(b,a)]} \\
\mathsf{s} & \overline{[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]}
\end{array}
\quad \rightsquigarrow \quad
\begin{array}{rc}
 & [([\bar{b},\bar{b}],[\bar{b},\bar{b}]),(b,a)] \\
\mathbf{t}\!\downarrow & \overline{[([\bar{b},\bar{b}],([\bar{b},\bar{b}],\mathbf{t})),(b,a)]} \\
\mathsf{ai}\!\downarrow & \overline{[([\bar{b},\bar{b}],([\bar{b},\bar{b}],[a,\bar{a}])),(b,a)]} \\
\mathsf{s} & \overline{[([\bar{b},\bar{b}],[([\bar{b},\bar{b}],\bar{a}),a]),(b,a)]} \\
\sigma\!\uparrow & \overline{[([\bar{b},\bar{b}],[(\bar{a},[\bar{b},\bar{b}]),a]),(b,a)]} \\
\sigma\!\downarrow & \overline{[([\bar{b},\bar{b}],[a,(\bar{a},[\bar{b},\bar{b}])]),(b,a)]} \\
\mathsf{ac}\!\downarrow & \overline{[(\bar{b},[a,(\bar{a},[\bar{b},\bar{b}])]),(b,a)]} \\
\mathsf{s} & \overline{[[(\bar{b},a),(\bar{a},[\bar{b},\bar{b}])],(b,a)]} \\
\mathsf{ac}\!\downarrow & \overline{[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]}
\end{array}
$$

Fig. 4. Example for type-B bureaucracy

$$
\begin{array}{rc}
 & [(\bar{b},\bar{b}),(b,a)] \\
\mathbf{t}\!\downarrow & \overline{[(\bar{b},(\bar{b},\mathbf{t})),(b,a)]} \\
\mathsf{ai}\!\downarrow & \overline{[(\bar{b},(\bar{b},[a,\bar{a}])),(b,a)]} \\
\mathsf{s} & \overline{[(\bar{b},[(\bar{b},\bar{a}),a]),(b,a)]} \\
\sigma\!\uparrow & \overline{[(\bar{b},[(\bar{a},\bar{b}),a]),(b,a)]} \\
\sigma\!\downarrow & \overline{[(\bar{b},[a,(\bar{a},\bar{b})]),(b,a)]} \\
\mathsf{s} & \overline{[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]}
\end{array}
\quad \rightsquigarrow \quad
\begin{array}{rc}
 & [(\bar{b},\bar{b}),(b,a)] \\
\sigma\!\uparrow & \overline{[(\bar{b},\bar{b}),(b,a)]} \\
\mathbf{t}\!\downarrow & \overline{[(\bar{b},(\bar{b},\mathbf{t})),(b,a)]} \\
\mathsf{ai}\!\downarrow & \overline{[(\bar{b},(\bar{b},[a,\bar{a}])),(b,a)]} \\
\mathsf{s} & \overline{[(\bar{b},[(\bar{b},a),\bar{a}]),(b,a)]} \\
\sigma\!\downarrow & \overline{[(\bar{b},[\bar{a},(\bar{b},a)]),(b,a)]} \\
\mathsf{s} & \overline{[[(\bar{b},\bar{a}),(\bar{b},a)],(b,a)]} \\
\sigma\!\uparrow & \overline{[[(\bar{a},\bar{b}),(\bar{b},a)],(b,a)]} \\
\sigma\!\downarrow & \overline{[[(\bar{b},a),(\bar{a},\bar{b})],(b,a)]}
\end{array}
$$

Fig. 5. Example for type-C bureaucracy

same a and $\bar{a}$ in the conclusion are "brought together" and disappear in an identity. The difference is that the derivation on the right contains two more applications of commutativity in which the two $\bar{b}$ are exchanged. But neither Formalism A nor Formalism B can identify the two. Let us call *Formalism C* a formalism that is able to avoid this kind of bureaucracy.

So far, none of the three formalisms mentioned above has been formalized as a deductive system.[4] Nonetheless, from the intuition given above one can translate the forced identifications in category theoretical terms. This is briefly done in Figure 6, which might be helpful for understanding the differences between the various kinds of bureaucracy. But the reader should be warned that this comparison os only very rough.[5] There are various issues which are still unclear an subject to future research. Most important are the questions: How does the

[4] But compare [BL05] for work on term calculi for Formalisms A and B.

[5] See also [Hug04] and [McK05] for work relating deep inference and categorical logic.

$$\boxed{\begin{array}{rl}
\text{deep inference} \simeq & \text{working in a syntactic category} \\[4pt]
\text{calculus of structures} \simeq & \text{free syntactic category} \\
& \text{(+ coherence for associativity and commutativity)} \\[4pt]
\text{Formalism A} \simeq & \text{free syntactic category} \\
& \ + \text{bifunctoriality of } -\wedge- \text{ and } -\vee- \\
& \text{(+ coherence for associativity and commutativity)} \\[4pt]
\text{Formalism B} \simeq & \text{free syntactic category} \\
& \ + \text{bifunctoriality of } -\wedge- \text{ and } -\vee- \\
& \ + \text{naturality of } \mathsf{s}, \mathsf{m}, \alpha\downarrow, \alpha\uparrow, \sigma\downarrow, \sigma\uparrow, \mathbf{t}\downarrow, \mathbf{t}\uparrow, \mathbf{f}\downarrow, \mathbf{f}\uparrow \\
& \text{(+ coherence for associativity and commutativity)} \\[4pt]
\text{Formalism C} \simeq & \text{free syntactic category} \\
& \ + \text{bifunctoriality of } -\wedge- \text{ and } -\vee- \\
& \ + \text{naturality of } \mathsf{s}, \mathsf{m}, \alpha\downarrow, \alpha\uparrow, \sigma\downarrow, \sigma\uparrow, \mathbf{t}\downarrow, \mathbf{t}\uparrow, \mathbf{f}\downarrow, \mathbf{f}\uparrow \\
& \ + \text{full coherence}
\end{array}}$$

Fig. 6. The bureaucracy-ABC vs. categorical logic

treatment of associativity and commutativity of the calculus of structures fit into the picture? How can we accomodate the units/constants? And, what are the axioms to which "full coherence" appeals?[6]

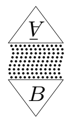
An alternative approach toward bureaucracy from a category theoretical viewpoint is based on n-categories and n-dimensional rewriting [Gui05]. Then proofs are three-dimensional objects, and bureaucracy is eliminated by isotopy.

Although formalisms A, B, C do not yet exist we can use proof nets to provide canonical representants of derivations in these formalisms. In Section 6 we will see the construction of such representants for formalisms A and C.

5 Derivation Nets

In this section we will see how derivations are translated into prenets. This is done by assigning to each rule (shallow or deep) a *rule net*:

$$\rho\,\frac{A}{B} \qquad \rightsquigarrow$$



where the linking is subject to certain side conditions which depend on the rule ρ. Figures 7, 8 and 9 show the rule nets for the rules of system SKS, as they are

[6] For classical logic there are now at least three different proposals for such an axiomatisation: [FP04], [DP04], and [LS05a]. But as we will see in Section 7, none of them can meet our needs.

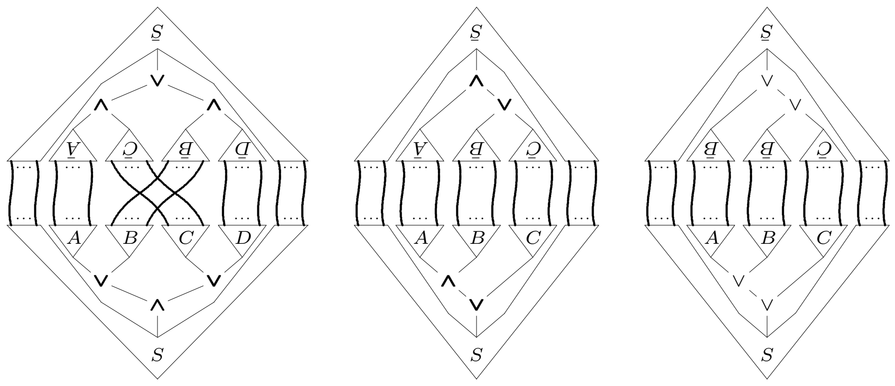


Fig. 7. The shape of m-nets, s-nets, and $\alpha\downarrow$-nets

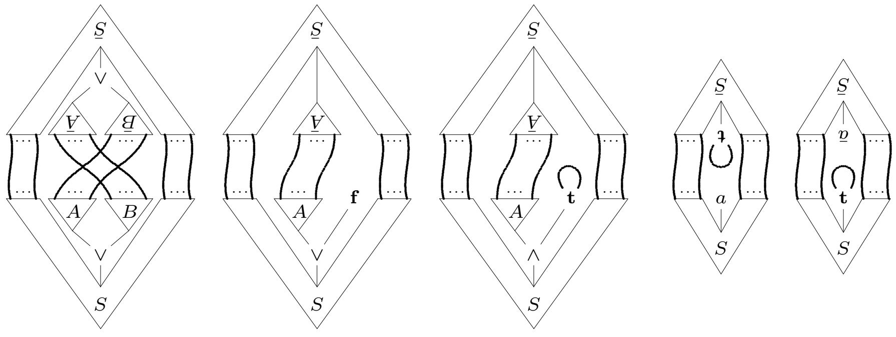


Fig. 8. The shape of $\sigma\downarrow$-nets, $\mathbf{f}\downarrow$-nets, $\mathbf{t}\downarrow$-nets, aw$\downarrow$-nets, and aw$\uparrow$-nets

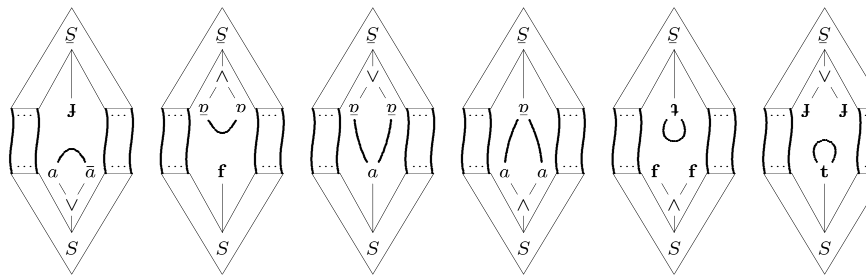


Fig. 9. The shape the nets for the rules ai$\downarrow$, ai$\uparrow$, ac$\downarrow$, ac$\uparrow$, nm$\downarrow$, and nm$\uparrow$

given in Figure 1. For the rules s, m, $\alpha\downarrow$, $\alpha\uparrow$, $\sigma\downarrow$, and $\sigma\uparrow$, it is intuitively clear what should happen: every atom in the premise is connected to its counterpart in the conclusion via an edge in in the linking; and there are no other edges. Note that the nets for $\alpha\downarrow$ and $\alpha\uparrow$ are the same; one written as the upside-down version of the other. The same holds for all other pairs of dual rules. For $\alpha\downarrow$, $\sigma\downarrow$, $\mathbf{f}\downarrow$, and

t↓ only one picture is shown, but for all other rules down- and up-version are given because it is instructive to see them next to each other. Note in Figure 9 also the difference between the atomic contraction and the nullary medial rules.

Now we can use cuts to plug rule nets together to get *derivation nets*, as it is shown in the upper left of Figure 10. Note that in derivation nets the "duality" between derivations

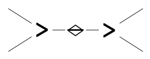
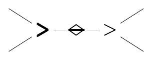
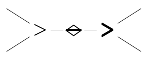
$$\begin{array}{c} A \\ \Delta \Big\Vert \mathsf{S} \\ B \end{array} \qquad \text{and} \qquad \begin{array}{c} \bar{B} \\ \bar{\Delta} \Big\Vert \mathsf{S} \\ \bar{A} \end{array}$$

disappears because both are represented by *the same* net.

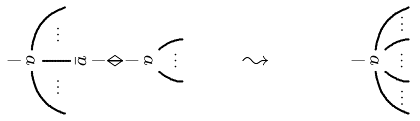
6 Cut Elimination is Losing Information

In this section we will see how cut elimination removes information, and that this can be bureaucratic as well as non-bureaucratic (i.e., essential) information.

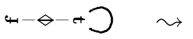
I will introduce three levels of cut elimination, that I call here "level-A", "level-C", and "level-X". For this, I need the following notion: An instance of a binary connective is called *heavy* if it is active in a medial or switch rule. The heavy instances of connectives are in boldface in Figure 7. All other instances of connectives are called *light*, i.e., all connectives appearing in the contexts and all connectives that are active in any other rule. For *level-A cut elimination* we allow the reduction of cuts on binary connectives only if both connectives are light. This means that

 ,  , and 

cannot be reduced. An atomic cut can only be reduced if at least one of the two cut-atoms has exactly one adjacent egde in the linking by which it is connected to another atom (its dual). I.e, we have:



Additionally we allow the reduction

 (7)

A derivation net is called *A-reduced* if no further cut elimination step under these restrictions is possible. Note that level-A reduction is clearly terminating, and it is also confluent because the problematic cuts are not allowed to reduce. Therefore, for every derivation net there is a unique A-reduced net. It should be obvious by now, that our goal is to establish the following claim:

Claim: *Two* SKS*-derivations yield the same A-reduced net if and only if they are identified by formalism A.*

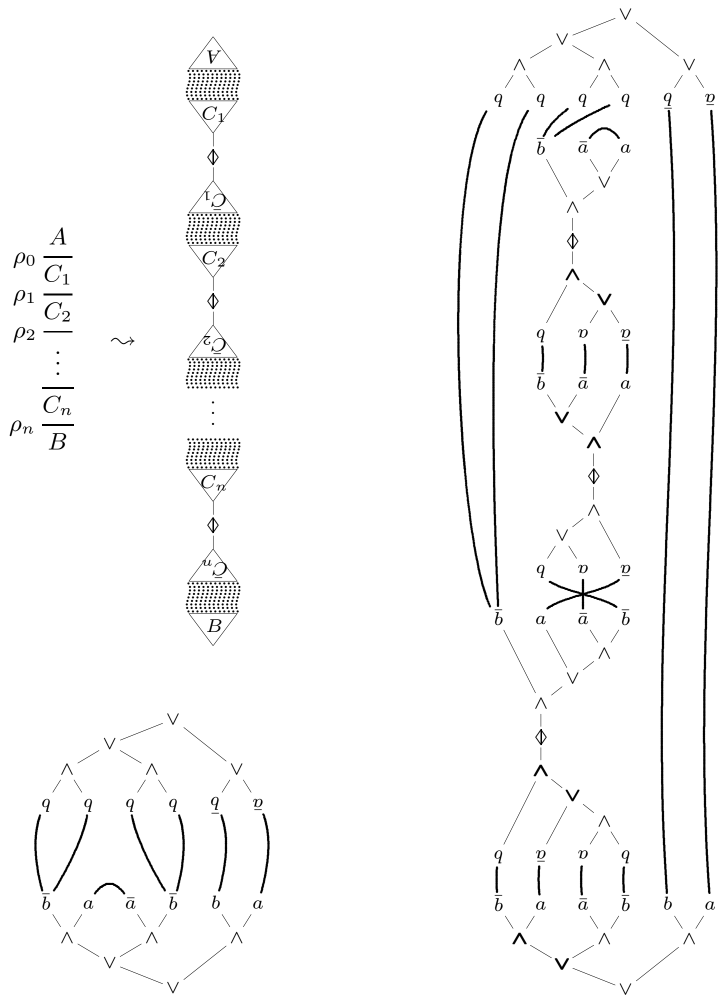
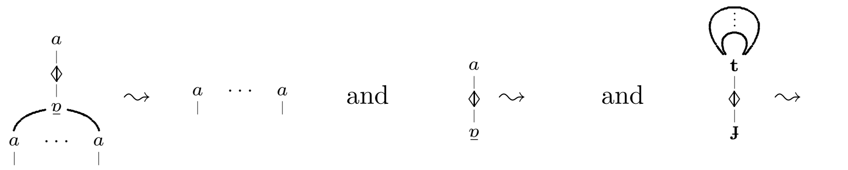


Fig. 10. Upper left: From derivations to derivation nets. **Right:** Example of an A-reduced net. **Lower left:** Result of applying level-C cut elimination to it.

In order to give a formal proof, it would be necessary to give a formal definition of formalism A. The reader will agree that at this point it would be an easy exercise to come up with a technical definition such that the claim holds. In fact, one could use the A-reduced nets as the defining criterion. However, a more interesting and not so trivial problem is to come up with a deductive formalism for A-reduced nets. Another problem is to establish the relation between the A-reduced nets and the term calculus for Formalism A presented in [BL05]. The conjecture would be that level-A cut elimination is "the same" as the term normalization of [BL05].

That there is a close relation between A-reduced nets and formalism A can be seen by observing that A-reduced nets identify the two derivations in (6), as well as any "merge" of Δ and Δ'. For example the net on the right of Figure 10 represents the two derivations in Figure 3. It is easy to see that this net cannot be obtained from the right derivation in Figure 4. This means that A-reduced nets are not able to identify the derivations that are identified by formalism B.[7]

Let us now define *level-C cut elimination*. Here we allow the reduction of all cuts on binary connectives, i.e., we forget the distinction between heavy and light instances of connectives. For cuts on atoms we additionally allow the reductions:



(8)

Note that the cuts that are problematic for confluency are still blocked. This means that for each net we have a unique *C-reduced* net. Examples for C-reduced nets are the two nets in (2). As the reader might verify, they are obtained from the two derivations in Figure 2, which is the reason why these two SKS-derivations are explicitely given. Another example is in the lower left of Figure 10 (which is obtained from the net on the right of that figure). As before, we have

Claim: *Two* SKS*-derivations yield the same C-reduced net if and only if they are identified by formalism C.*

And, as before, coming up with a formal definition of formalism C such that the claim holds is an easy and uninteresting exercise. The difficult question is: What are the right axioms that freely generate the category of C-reduced nets? Or, equivalently, what is a minimal set of equations to be enforced on derivations such that the equivalence classes are in bijection with the C-reduced nets?

Finally, we speak of *level-X cut elimination*, if there are no restrictions on the reduction. A net is *X-reduced* if it contains no cuts. Hence, the X-reduced prenets are exactly the $\mathbb{N}$-prenets of [LS05b]. For example, the right net in (2) can be obtained by reducing the cut from the left one. This means that X-reduced nets would identify the two derivations in Figure 2. We can safely assume that this goes beyond mere "bureaucracy elimination". With level-X cut elimination we not only eliminate "essential" information, we also leave the realm of confluency (see [LS05b] for an explanation).

It should be mentioned that the X-reduced nets are essentially the same as Buss' *logical flow graphs* [Bus91,Car97].

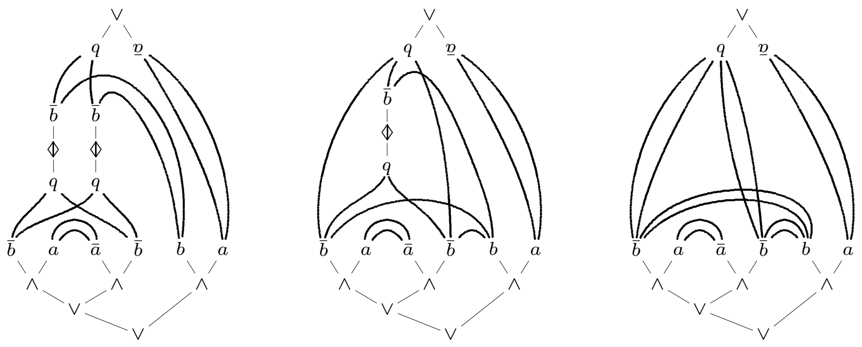
[7] Note that with A-reduced nets we make already some identifications that the purist would put into the realm of formalism B, or even formalism C. For "pure formalism A" it would be necessary to make all active conectives heavy and only the ones in the context light. Further, we would have to disallow (7). I deviated her from the "purist's way" because the coherence laws for symmetric monoidal categories are so natural.

7 Some remarks on the induced categories

We can define two categories: Let **Pre** be the category whose objects are the formulae and whose arrows between formulae A and B are the C-reduced prenets with $\Gamma = \bar{A}, B$. The category **Deri** is the wide subcategory of **Pre** in which the Hom-sets contain only those C-reduced prenets which are obtained from an SKS-derivation in the way described in the previous sections. Note that the X-reduced nets do not form a category, because composition defined via cut elimination is not associative. The A-reduced nets do also not form a category because the identity nets do not behave as identities.

The category **Deri** can be called "Boolean" in the sense of [LS05a]. The forgetful functor from the category of (small) categories to the category of posets maps it to a Boolean algebra. For the category **Pre** this is not the case because it contains morphisms that do not correspond to implications in Boolean logic.

I cannot give here a full characterisation of the two categories **Pre** and **Deri**, but I will compare them to the three different axiomatisations given in [FP04], [DP04], and [LS05a]. All of them have in common that the Hom-sets are equipped with an idempotent semigroup structure. This semigroup structure is also present for **Pre** and **Deri**, but it is *not* idempotent. In the case of **Pre** the sum of two nets is given by their union. This is best understood by seeing an example. Let f be the left net in (2) and g the right one. Then we can form $f+f$, $f+g$, and $g+g$ as follows:



In the case of **Deri** we can also form the "sum" of two derivations by using contraction, i.e., the two rules

$$\mathsf{c}\!\downarrow \frac{S[A,A]}{S\{A\}} \qquad \text{and} \qquad \mathsf{c}\!\uparrow \frac{S\{A\}}{S(A,A)} \quad ,$$

which are both derivable in SKS (see [BT01] for details), and the rule

$$\mathsf{mix} \frac{S(A,B)}{S[A,B]} \quad ,$$

which is also derivable in SKS, for example via

$$
\begin{array}{rc}
 & S(A,B) \\ \cline{2-2}
\mathbf{f}{\downarrow} & S([A,\mathbf{f}],B) \\ \cline{2-2}
\mathsf{aw}{\downarrow} & S([A,\mathbf{t}],B) \\ \cline{2-2}
\mathsf{s} & S[A,(\mathbf{t},B)] \\ \cline{2-2}
\mathbf{f}{\uparrow} & S[A,B]
\end{array}
\quad .
$$

For any two derivations Δ_1, Δ_2 from A to B we can now form their sum by taking

$$
\begin{array}{rc}
 & A \\ \cline{2-2}
\mathsf{c}{\uparrow} & (A,A) \\
{\scriptstyle (\Delta_1,\Delta_2)} & \Big\Vert {\scriptstyle \mathsf{SKS}} \\
 & (B,B) \\ \cline{2-2}
\mathsf{mix} & [B,B] \\ \cline{2-2}
\mathsf{c}{\downarrow} & B
\end{array}
\quad ,
$$

where (Δ_1,Δ_2) is some "merge" of Δ_1 and Δ_2; compare (6). Note, that this sum of derivations could also be obtained in a different way, for example by first mixing (A,A) and then taking $[\Delta_1,\Delta_2]$, or by using a different derivation for mix. However, the important observation to make here is that no matter which one we choose, the translation into a C-reduced prenet yields the same result for all of them; and we obtain the same result if we first translate the derivations Δ_1 and Δ_2 into C-reduced prenets, and then taking their sum as nets (as described above). Hence, the semigroup structure on the Hom-sets is the same for **Pre** and **Deri**.

Furthermore, we can equip the Hom-sets of our categories with a partial order, defined by cut elimination: We say $f \le g$ if g is obtained from f by eliminating some of the remaining cuts[8], as it is the case in our example above for f and g. Then we also have $f+f \le f+g \le g+g$. The important observation about the semigroup and the partial order structure is, that *they are independent*. Although this seems to be natural from the viewpoint of our nets, it is not the case in the "classical categories" of [FP04] which are based on the proof nets in [Rob03]. In a "classical category" the sum-of-proofs-semigroup structure and the cut-elimination-partial-order structure on the Hom-sets determine each other uniquely via $f \le g$ iff $f+g=g$. (In [DP04] and [LS05a] there is also a partial order structure on the Hom-sets, simply because the semigroup structure is idempotent. But this partial order structure has nothing to do with cut elimination, simply because everything is *a priory* cut-free.)

The category **Pre** follows quite closely the axiomatisation given in [LS05a]: it is *-autonomous (with weak units), it has monoids and comonoids, and it is

[8] Even if this process is not confluent, we stay in the realm of C-reduced nets.

"graphical". But is does not obey the equation

$$\begin{tikzcd}
A \vee A \arrow[r, "\Delta_{A \vee A}"] \arrow[d, "\nabla_A"'] & (A \vee A) \wedge (A \vee A) \arrow[d, "\nabla_A \wedge \nabla_A"] \\
A \arrow[r, "\Delta_A"'] & A \wedge A
\end{tikzcd} \tag{9}$$

However, the two maps $A \vee A \to A \wedge A$ in (9) are ordered according to the cut-elimination-partial-order defined above, as it is the case in [FP04].

For the category **Deri** it is almost the same. The difference is that I have no proof showing that it is *-autonomous. Furthermore, I do not know whether **Deri** is closed under cut-elimination: Let Δ be an SKS derivation and let f_Δ be its corresponding C-reduced net. Now let f' be a net obtained from f_Δ by reducing some of the remaining cuts. Is there an SKS-derivation Δ' corresponding to f'?

Solving these two open problems (and, in case of a negative answer, find a better deep inference deductive system for classical logic) is an important pre-requisite for starting to look for a decent geometrical correctness criterion for proof nets. Only with a good behaved deductive system one can ask for a "sequentialization theorem" for proof nets.

References

[BL05] Kai Brünnler and Stéphane Lengrand. On two forms of bureaucracy in derivations. In *Structures and Deduction 2005 (Satellite Workshop of ICALP'05)*, 2005.

[Brü03a] Kai Brünnler. *Deep Inference and Symmetry for Classical Proofs*. PhD thesis, Technische Universität Dresden, 2003.

[Brü03b] Kai Brünnler. Minimal logic in the calculus of structures. note, 2003.

[BT01] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 347–361. Springer-Verlag, 2001.

[Bus91] Samuel R. Buss. The undecidability of k-provability. *Annals of Pure and Applied Logic*, 53:72–102, 1991.

[Car97] Alessandra Carbone. Interpolants, cut elimination and flow graphs for the propositional calculus. *Annals of Pure and Applied Logic*, 83:249–299, 1997.

[DG04] Pietro Di Gianantonio. Structures for multiplicative cyclic linear logic: Deepness vs cyclicity. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *Computer Science Logic, CSL 2004*, volume 3210 of *Lecture Notes in Computer Science*, pages 130–144. Springer-Verlag, 2004.

[DP04] Kosta Došen and Zoran Petrić. *Proof-Theoretical Coherence*. KCL Publications, London, 2004.

[FP04] Carsten Führmann and David Pym. Order-enriched categorical models of the classical sequent calculus. To appear in *Journal of Pure and Applied Algebra*, 2004.

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

[Gir91] Jean-Yves Girard. A new constructive logic: Classical logic. *Mathematical Structures in Computer Science*, 1:255–296, 1991.

[Gir96] Jean-Yves Girard. Proof-nets : the parallel syntax for proof-theory. In Aldo Ursini and Paolo Agliano, editors, *Logic and Algebra*. Marcel Dekker, New York, 1996.

[GS02] Alessio Guglielmi and Lutz Straßburger. A non-commutative extension of MELL. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2002*, volume 2514 of *LNAI*, pages 231–246. Springer-Verlag, 2002.

[Gug02] Alessio Guglielmi. A system of interaction and structure. To appear in *ACM Transactions on Computational Logic*, 2002.

[Gug04a] Alessio Guglielmi. Formalism A. note, April 2004.

[Gug04b] Alessio Guglielmi. Formalism B. note, December 2004.

[Gui05] Yves Guiraud. The three dimensions of proofs. In *Structures and Deduction 2005 (Satellite Workshop of ICALP'05)*, 2005.

[Hug04] Dominic Hughes. Deep inderence proof theory equals categorical proof theory minus coherence. preprint, 2004.

[HvG03] Dominic Hughes and Rob van Glabbeek. Proof nets for unit-free multiplicative-additive linear logic. In *18th IEEE Symposium on Logic in Computer Science (LICS 2003)*, pages 1–10, 2003.

[KM71] Gregory Maxwell Kelly and Saunders Mac Lane. Coherence in closed categories. *Journal of Pure and Applied Algebra*, 1:97–140, 1971.

[LS05a] François Lamarche and Lutz Straßburger. Constructing free Boolean categories. In *Proceedings of the Twentieth Annual IEEE Symposium on Logic in Computer Science (LICS'05)*, 2005.

[LS05b] François Lamarche and Lutz Straßburger. Naming proofs in classical propositional logic. In Paweł Urzyczyn, editor, *Typed Lambda Calculi and Applications, TLCA 2005*, volume 3461 of *Lecture Notes in Computer Science*, pages 246–261. Springer-Verlag, 2005.

[McK05] Richard McKinley. Classical categories and deep inference. In *Structures and Deduction 2005 (Satellite Workshop of ICALP'05)*, 2005.

[Rob03] Edmund P. Robinson. Proof nets for classical logic. *Journal of Logic and Computation*, 13:777–797, 2003.

[SL04] Lutz Straßburger and François Lamarche. On proof nets for multiplicative linear logic with units. In Jerzy Marcinkowski and Andrzej Tarlecki, editors, *Computer Science Logic, CSL 2004*, volume 3210 of *LNCS*, pages 145–159. Springer-Verlag, 2004.

[SS03] Charles Stewart and Phiniki Stouppa. A systematic proof theory for several modal logics. Technical Report WV-03-08, Technische Universität Dresden, 2003. To appear in proceedings of *Advances in Modal Logic 2004*, published by King's College Publications.

[Sto04] Finiki Stouppa. The design of modal proof theories: the case of S5. Master's thesis, Technische Universität Dresden, 2004.

[Str02] Lutz Straßburger. A local system for linear logic. In Matthias Baaz and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2002*, volume 2514 of *LNAI*, pages 388–402. Springer-Verlag, 2002.

[Str03] Lutz Straßburger. *Linear Logic and Noncommutativity in the Calculus of Structures*. PhD thesis, Technische Universität Dresden, 2003.

[Tiu05] Alwen Fernanto Tiu. A local system for intuitionistic logic: Preliminary results. preprint, 2005.