To appear in: *Theoretical Computer Science (1994)*, Proceedings MFPS8.

On the π-Calculus and Linear Logic

G. Bellin *　　P. J. Scott †

May 30, 1994

Abstract

We detail Abramsky's "proofs-as-processes" paradigm for interpreting classical linear logic (CLL) [13] into a "synchronous" version of the π-calculus recently proposed by Milner [27, 28]. The translation is given at the abstract level of proof structures. We give a detailed treatment of information flow in proof-nets and show how to mirror various evaluation strategies for proof normalization. We also give Soundness and Completeness results for the process-calculus translations of various fragments of CLL. The paper also gives a self-contained introduction to some of the deeper proof-theory of CLL, and its process interpretation.

Contents

*Research Supported by Esprit grant BRA Project 3245.

†Research supported by an operating grant from the Natural Sciences and Engineering Research Council of Canada, FCAR Team Grants from Québec, and a Bilateral Exchange Grant from NSERC and the Royal Society of Britain

1 Introduction

Milner's π-calculus [25, 26] is a recent addition to a large and active literature on the foundations of *concurrent computation*. These theories attempt to analyze and clarify the world of concurrently communicating processes (and associated programming languages) in much the same way as lambda calculus and other models of computation have done for the sequential world [26].

In a different direction, Jean-Yves Girard [13, 16, 17, 18] has instituted the rapidly growing area of *linear logic*, a radical modification of traditional logic which appears to have strong connections with theoretical computer science. In several publications [16, 15] Girard has suggested that linear logic should have deeper connections with parallelism and concurrent computation. This suggestion was taken up more formally by Abramsky [1] in an influential series of lectures, and in some unpublished work of Robin Milner [24].

The Abramsky view is essentially a modification of the familiar formulae-as-types (Curry-Howard) isomorphism: instead of proofs being functions (qua lambda terms), Abramsky views proofs as processes (e.g. π-calculus or CCS terms). Processes are thought of as communicating through distinguished ports or channels, named by free variables. The key observation is that proof-theoretical communication (i.e. the Cut rule in the Gentzen formalism) is modelled by communication along a private internal channel (i.e. *hiding*) in the process-calculi world. It should follow that the *dynamics* of logical computation, namely the cut-elimination or normalization process for proofs, be reflected in the rewriting theory for process algebra terms.

Alas, this is precisely the point where the concurrency and logic worlds begin to diverge. For example, the fundamental case of cut-elimination requires *modular* or *contextual* rewriting (since we may need to eliminate a cut embedded high-up in a proof tree.) Most experts on concurrency theory are simply unwilling to allow rewriting within *all* contexts, particularly those involving nondeterministic choice $+$; typically an interaction with a process $P + Q$ chooses one of the components, and the other component is immediately destroyed. The question is: to what extent can we represent logical computation within the accepted concurrency world?

Robin Milner [27] recently developed a version of the π-calculus (the *synchronous π-calculus*) purposely supporting some of the logical rewriting envisioned by Abramsky. In this paper, we analyze the Abramsky view in detail for the synchronous π-calculus. We shall translate proofs (and certain normalization strategies) for the three important levels of linear logic: the multiplicatives, the additives, and the exponentials. In the case of multiplicative linear logic, the π-calculus translation provides a lock-step simulation of proof net reduction. As soon as the additive and exponential connectives are included, again problems of rewriting in contexts begin to appear (even for the synchronous π-calculus). We show how we may soundly and completely reflect the proof theory (though not necessarily lock-step simulation) by different methods:

- Modify the π-calculus reduction strategies to more closely mimic those inherent in logic (e.g. (i) by introducing guarding of terms [25] or (ii) by a version of Girard's theory of *slicing* of proof nets [13, 6]).

- Modify linear logic evaluation strategies to take into account the extant theory of π-calculus (e.g. restriction to Geometry of Interaction-style evaluation strategies [17, 18])

One of the theses of this work is that the Abramsky-style translations (of linear logic) into the process world actually have less to do with logic than one might think: they are essentially only about the abstract pluggings in proof structures [13, 16, 12], and we formulate many of our results in this more general setting. This phenomenon may be already expected by experts in concurrency theory, who view process algebras as a theory of "hand-shaking" protocols (which have nothing to do with logic) but it came as a surprise to us. What seems particularly curious–given this "alogical" nature of π-calculus–is that much of the theory (as described below) is nonetheless coherent (and complete) for various logical proof reduction (normalization) strategies.

A second theme of this paper is that information flow in cut elimination is related to sending/receiving protocols in π-calculus. In Section 5, this intuition is made precise in a detailed analysis of information flow in pure nets [11, 22, 8] which is a theory of graphical networks for representing untyped lambda terms in the style of proof nets for linear logic. We shall show how to dynamically orient a proof-net (assigning the symbols I (for input) and O (for output)) in a manner coherent with respect to introduction and elimination rules for natural deduction, and cut-elimination (normalization). This permits a systematic analysis of orientation and information flow in the Abramsky π-translation, obviating (for example) the need for bidirectional buffers.

Finally, the paper contains a survey of more advanced proof-theory for linear logic, including additive boxes and Girard's theory of slicing, and how to represent these notions in the π-calculus.

At this point, we should mention a few papers on related themes. The paper of Monteiro [29] gives a lock-step simulation of proof nets for multiplicative linear logic into a version of CSP, although the author does not handle the additives or exponentials. This paper is the first example of the kind of analysis considered here (apparently independent of Abramsky [2]) of a sound and faithful translation of logical deduction in a language of concurrency. The paper of Dale Miller [23] is based on a logic programming view of the π-calculus: Miller codes the π-calculus as a *theory* in linear logic, and discusses how techniques of logic programming (proof search, etc.) are useful in understanding some of the metatheory of the π-calculus. This paper has no direct connection with the Abramsky program of proofs-as-processes, although some of the syntactical considerations are mildly similar.

2 The π-Calculus

The following version of the π-calculus [24, 25] has recently been suggested by Milner [27] for its possible connections to linear logic, a connection we shall develop in more detail below.

2.1 Basic Synchronous π-calculus

There is an infinite family $\mathcal{X}$ of variables called *names*. We shall denote names by $x, y, z, \ldots \in \mathcal{X}$ and vectors of names by $\vec{x}$ or $\tilde{x}$. The π-calculus describes certain basic entities known as *processes*, having the following syntactic forms:

$$\bullet \bigm| (\nu x)P \bigm| P \parallel Q \bigm| \pi P$$

where:

(i) The prefixes π are formal expressions of the form $\overline{x}\langle\vec{y}\rangle$ or $x(\vec{y})$, where x is a name, and $\vec{y}$ is a vector of names, possibly of length zero. We say P is the *scope* of prefix π in the process πP.

(ii) νx and $(\vec{y})$ are name-binding operators with the obvious scopes. Thus in the expression $(\nu x)P$, the name x is bound and the scope of νx is P; the operation ν is known as *hiding*: the variable x in the process $(\nu x)P$ is said to be a hidden name. Similarly, in the expression $x(\vec{y})P$, the names $\vec{y}$ are bound (and x is free) and P is the scope.[1]

[1]More precisely, we may define the set of free and bound names in processes P (= $\mathrm{fn}(P)$ and $\mathrm{bn}(P)$, respectively) as follows: (a) $\mathrm{fn}(\bullet) = \mathrm{bn}(\bullet) = \emptyset$. (b) $\mathrm{fn}(P \parallel Q) = \mathrm{fn}(P) \cup \mathrm{fn}(Q)$; $\mathrm{bn}(P \parallel Q) = \mathrm{bn}(P) \cup \mathrm{bn}(Q)$. (c) $\mathrm{fn}(\overline{x}\langle\vec{y}\rangle P) = \{\overline{x}, \vec{y}\} \cup \mathrm{fn}(P)$; $\mathrm{bn}(\overline{x}\langle\vec{y}\rangle P) = \mathrm{bn}(P)$. (d) $\mathrm{fn}(x(\vec{y})P) = \mathrm{fn}(P) \cup \{x\}$; $\mathrm{bn}(x(\vec{y})P) = \mathrm{bn}(P) \cup \{\vec{y}\}$. (e) $\mathrm{fn}((\nu x)P) = \mathrm{fn}(P) \setminus \{x\}$; $\mathrm{bn}((\nu x)P) = \mathrm{bn}(P) \cup \{x\}$.

The intended interpretation is that π-calculus describes a theory of processes concurrently communicating through distinguished ports or channels (think of telephone systems or a computer network). The intended meaning of names [24] is that they represent *communication channels*, i.e. ports for the communicating behaviour of these interactive processes. A fundamental such operation is the "hand-shaking" protocol, in which one agent's channel identifies itself to another agent's channel, and the receiver signifies it is ready to receive the communication. This is encapsulated in the two syntactic operations $\overline{x}\langle\vec{y}\,\rangle P$, which means "send the names $\vec{y}$ along the channel x and then do process P ", and $x(\vec{y}\,)P$ which means: "receive along x arbitrary names for the bound names $\vec{y}$". Another fundamental operation is "hiding", in which a channel is declared private and so is inaccessible to the outside world. Finally $P \parallel Q$ denotes the process in which P and Q are acting concurrently (in parallel) while $\bullet$ denotes the nil process.

The following rules govern the behaviour of π-calculus terms. We assume there is a congruence relation $\equiv$ on π-terms which satisfies the following additional properties:

1. $\omega_1\omega_2 P \equiv \omega_2\omega_1 P$ provided no free variables become bound, nor any bound variables become free.
2. $\omega(P \parallel Q) \equiv \omega P \parallel Q$, provided $\mathrm{bn}(\omega) \cap \mathrm{fn}(Q) = \emptyset$.
3. $\parallel$ –as a binary operation on processes– satisfies the axioms of a commutative monoid with unit $\bullet$.
4. $(\nu x)\bullet \equiv \bullet$.
5. $(\nu x)(\nu y)P \equiv (\nu y)(\nu x)P$.

As usual, we identify processes which are identical except for change of bound variables (α-conversion). We often omit writing the process $\bullet$ (cf. [25]), writing $\bar{x}\langle\vec{y}\,\rangle$ in place of $\bar{x}\langle\vec{y}\,\rangle\bullet$.

We make the π-calculus into a rewriting system by assuming the following notion of basic 1-step reduction:

$$\bar{x}\langle\vec{y}\,\rangle P \parallel x(\vec{z})Q \succ P \parallel Q[\vec{y}/\vec{z}]$$

where $Q[\vec{y}/\vec{z}]$ denotes the simultaneous substitution of the names y_i for the (possibly free) names z_i in Q. We consider the rewriting theory modulo $\equiv$, so that for all contexts $\mathcal{C}$ we have: $P \succ Q \Longrightarrow \mathcal{C}[P] \succ \mathcal{C}[Q]$

This version of the π-calculus allows some unexpected rewritings (modulo $\equiv$); for example: $\bar{x}\langle\vec{y}\rangle x(\vec{z})P \succ P[\vec{y}/\vec{z}]$. *Proof:* $\bar{x}\langle\vec{y}\rangle x(\vec{z})P \equiv \bar{x}\langle\vec{y}\rangle(\bullet \parallel x(\vec{z})P) \equiv (\bar{x}\langle\vec{y}\rangle\bullet \parallel x(\vec{z})P) \succ P[\vec{y}/\vec{z}]$.

2.2 Full Synchronous π-calculus

We add to the syntax of the basic theory above, an additional process formation rule : finite summation. That is, the processes now have the following form:

$$\bullet \;\Big|\; (\nu x)P \;\Big|\; P \parallel Q \;\Big|\; \pi P \;\Big|\; P + Q$$

$P + Q$ represents a process able to take part in one (but only one) of the alternatives P, Q ; however, the choice is not made by the process, but by the environment at the time of a particular interaction. We add to the previous congruence rules for $\equiv$ the following rule

- The operation $+$ –considered as an operation on processes– satisfies the axioms of a commutative monoid with identity $\bullet$.

Moreover, we add to the reduction rules above the following additional basic rewrite rule:

$$(P + \bar{x}\langle\vec{z}\rangle Q) \parallel ((x(\vec{y})R) + S) \;\succ\; Q \parallel R[\vec{z}/\vec{y}]$$

We do not assume $\succ$ is a congruence with respect to $+$ –i.e. we do not assume rewriting under a $+$. It is worth remarking that the choice operator $+$ above has a powerful effect: it destroys all "side processes" P and S not actually involved in the interaction [2]. This is one of the main problematic operators in understanding π-calculus from the logical viewpoint.

[2] In particular, S is arbitrary, so may itself be of the form $x(\vec{y})S'$ So, assuming commutativity of $+$, basic interactions of the above form could sometimes (up to $\equiv$) arbitrarily pick either component when rewriting a summand.

Two important notions from the ordinary π-calculus are *guarding* and *distributivity*. Guarding is a method of forcing certain communications to occur in a given order, thus imposing a restriction on the evaluation strategies available for π-calculus terms. Syntactically, a dot after a prefix, as in $\omega.P$, denotes a process in which communication must occur with ω *before* any communication can occur within P (so in a sense the prefix ω "guards" P against contact with ambient processes, as well as preventing internal communication within P until ω is discharged). Writing the dot "." has no other formal status than to denote this ordering of evaluation. If we speak of synchronous π-calculus with guarding, we shall mean (unless otherwise stated) that all processes can have prefix sequences which may be mixtures of either guarded or unguarded ones.

Rewriting theory in the presence of guarding is a slight modification of that of the basic synchronous calculus above. We modify the rules mentioned in section 2.1, as follows:

- We can permute unguarded prefixes up to $\equiv$, with the usual provisos on free and bound variables, with the following restriction: properties (1) and (2) of the congruence $\equiv$ do not apply if one of the prefixes ω_i or ω is a guarding prefix.
- A subterm $\omega.P$ can *only* be reduced via communication with ω, not through reduction within P.

In particular, under guarding rewriting is not *contextual* or *compositional*.

Example: The ordinary process term $\bar{x}\langle z\rangle P \parallel x(y)Q$ rewrites as follows:

$$\bar{x}\langle z\rangle P \parallel x(y)Q \succ P \parallel Q[z/y]$$

whereas the guarded term $u(x).[\bar{x}\langle z\rangle P \parallel x(y)Q]$ is at the moment inert: it cannot have any internal action and can only communicate through the variable u.

Milner [25] uses guarding to constrain evaluation strategies (e.g. in simulating the lazy λ- calculus in the π-calculus). Similarly, in section 6.2 we shall use guarding to mirror certain restricted evaluation strategies in linear logic (needed for our completeness theorems for additive proof net reduction).

Distributivity (of $\parallel$ and unguarded prefixes) with respect to $+$ is discussed in [25] and other references referred to there. The main feature (from our viewpoint) is that distributive laws are needed to mirror the commutative reductions of linear logic (cf [13]).

3 Proofs-as-Processes

In this section we introduce a version of the Abramsky translation [2] mapping proofs in linear logic into process calculi, and discuss Soundness and Completeness Theorems for this translation. As mentioned in the Introduction, our treatment is an *adaptation* of the original Abramsky work to the synchronous π-calculus, a calculus better suited to this kind of analysis. The idea is to assign to a proof annotated with free variables, a process term whose free variables are exactly the variables in the conclusion of the proof:

$$\begin{array}{c} \vdots \\ \vdash x_1 : A_1, \ldots, x_n : A_n \end{array} \qquad \rightsquigarrow \qquad P x_1, \ldots, x_n$$

One may think of the free variables $x_1, \ldots, x_n$ as "communication ports" in an interface connecting a process to its surroundings.

We assume the reader is familiar with the usual presentations of classical linear logic (CLL) by one-sided sequents [1, 13]. We consider, in separate sections below, an involutive negation $(\;)^\perp$ along with the three levels of CLL connectives emphasized by Girard [13], which are pairwise related by de Morgan duality: (i) *multiplicatives*: the connectives $\{\otimes, \parr\}$, i.e. tensor and dual tensor (= par), (ii) *additives* : the connectives $\{\&, \oplus\}$, i.e. product and coproduct , (iii) *exponentials*: the connectives $\{!, ?\}$, which are storage operators. We write MLL for the multiplicative fragment, MALL for the multiplicative and additive fragment, and CLL for the full theory. Note that we ignore the role of the units in all that follows. For simplicity, we also consider only atomic axiom links.

Much of the proof theory used in this paper is standard; further details for the case of linear logic are contained in Girard's original paper [13], Troelstra's recent book [32], as well as in [6, 7], [12], etc.

3.1 The Abramsky Translation: the multiplicatives

Logical Rule	**π-translation**
$\vdash x : A, y : A^{\perp}$	$Ixy \;=\; x(a)\overline{y}\langle a\rangle$
$\dfrac{\begin{array}{cc} \vdots\, F & \vdots\, G \\ \vdash \vec{w} : \Gamma, x : A & \vdash \vec{v} : \Delta, y : B \end{array}}{\vdash \vec{w} : \Gamma, \vec{v} : \Delta, z : A \otimes B}\,\otimes$	$\bigotimes_{z}^{x,y}(F,G)\vec{w}\vec{v}z \;=\; \nu xy(\overline{z}\langle xy\rangle(F\vec{w}x \parallel G\vec{v}y))$
$\dfrac{\begin{array}{c} \vdots\, F \\ \vdash \vec{w} : \Gamma, x : A, y : B \end{array}}{\vdash \vec{w} : \Gamma, z : A ⅋ B}\,⅋$	$(⅋_{z}^{x,y})(F)\vec{w}z \;=\; z(xy)F\vec{w}xy$
$\dfrac{\begin{array}{cc} \vdots\, F & \vdots\, G \\ \vdash \vec{u} : \Gamma, x : C & \vdash \vec{v} : \Delta, y : C^{\perp} \end{array}}{\vdash \vec{u} : \Gamma, \vec{v} : \Delta}\,Cut$	$Cut^{z}(F,G)\vec{u}\vec{v} = \nu z(F\vec{u}[z/x] \parallel G\vec{v}[z/y])$

It is worth remarking that this translation makes two choices: we choose to interpret $\otimes$ uniformly as a *sender* and ⅋ uniformly as a *receiver*. Following the terminology of Abramsky and Milner, the operator I in the translation of the axiom is known as an *axiom buffer*. Note that axiom buffers also translate positive atoms as receivers and negative atoms as senders. A full discussion of information flow and these choices is in Section 5 below. Superscripts on the π-terms $\bigotimes$, ⅋, Cut denote bound variables.

The Cut Algebra for MLL

The following equations represent, in functional (combinator) form, the cut-elimination reductions in MLL.

Symmetric Reductions

$$
\begin{array}{llcl}
(1) & Cut^{x}(F,G) & = & Cut^{x}(G,F) \\
(2) & Cut^{x}(Fx, Ixy) & \succ & F[y/x] \\
(3) & Cut^{z}(\bigotimes_{z}^{x,y}(Fx,Gy), ⅋_{z}^{x,y}(Hxy)) & \succ & Cut^{y}(Gy, Cut^{x}(Fx,Hxy))
\end{array}
$$

$$(4) \qquad \equiv \quad Cut^x(Fx, Cut^y(Hxy, Gy))$$

Commutative Reductions

$$(5) \qquad Cut^x(\wp_v^{c,d} Fxcd, Gx) \quad = \quad \wp_v^{c,d} Cut^x(Fxcd, Gx)$$

$$(6) \qquad Cut^x(\bigotimes_v^{c,d}(Fc, Gdx), Hx) \quad = \quad \bigotimes_v^{c,d}(Fc, Cut^x(Gdx, Hx))$$

where in the first and second commutative reduction equations, $\wp$ and $\otimes$ do not react with x.

Remark

Note that in the term $\wp_v^{c,d} Cut^x(Fxc, Gxd)$, Cut and Par obviously do not permute. In order to permute inferences, one needs the additional information that c, d occur in the same branch of the proof tree, as in (5) above.

Theorem 1 (Soundness) *Let $\mathcal{D}$ be a proof in MLL and let $\pi(\mathcal{D})$ be its π-calculus translation.*

(i) If $\mathcal{D} \succ_1 \mathcal{D}'$ by a 1-step symmetric reduction in the cut-elimination process, then $\pi(\mathcal{D}) \succ_1 \pi(\mathcal{D}')$ in the synchronous π-calculus.

(ii) If $\mathcal{D} \succ_1 \mathcal{D}'$ by a 1-step commutative reduction in the cut-elimination process, then $\pi(\mathcal{D}) \equiv \pi(\mathcal{D}')$ in the synchronous π-calculus.

Proof: By induction, following the steps in cut-elimination. ∎

From part (ii) in the above theorem, we see that the π translation really acts on proofs modulo the order of the rules, thus on proof *nets* [13]. This theorem will be generalized in Sections 4, 5 below to the case of proof structures. We shall also discuss the *completeness* (or in categorical language, the *local fullness*) of the π-calculus translation.

It is important to observe that cut-elimination is a *modular* or *compositional* property: a reduction $\mathcal{D} \succ_1 \mathcal{D}'$ may occur high up in the proof tree, not necessarily at the bottom node. This requires that reduction in the associated π-terms must be a congruence, i.e reduction must be *contextual.* This will be the case for the π-terms that arise in the MLL translation. But there will be problems with the Additives and Exponentials, as we shall see below.

3.2 The Abramsky Translation: the additives

Logical Rule | **π-translation**

$$\frac{\begin{array}{c}\vdots\, P\\ \vdash \vec{w}:\Gamma, x:A\end{array}}{\vdash \vec{w}:\Gamma, z:A\oplus B}\oplus \qquad L_z^x(P)\vec{w}z \;=\; (\nu x)(z(uv)\bar{u}\langle x\rangle P_{\vec{w}x})$$

$$\frac{\begin{array}{c}\vdots\, Q\\ \vdash \vec{w}:\Gamma, y:B\end{array}}{\vdash \vec{w}:\Gamma, z:A\oplus B}\oplus \qquad R_z^y(Q)\vec{w}z \;=\; (\nu y)(z(uv)\bar{v}\langle y\rangle Q_{\vec{w}y})$$

$$\frac{\begin{array}{c}\vdots\, P\\ \vdash \vec{w}:\Gamma, x:A\end{array}\quad \begin{array}{c}\vdots\, Q\\ \vdash \vec{w}:\Gamma, y:B\end{array}}{\vdash \vec{w}:\Gamma, z:A\&B}\,\& \qquad \&_z^{xy}(P,Q)\vec{w}z \;=\; \nu(uv)\bar{z}\langle uv\rangle[u(x)P_{\vec{w}x}+v(y)Q_{\vec{w}y}]$$

The Cut Algebra: Additives

The following equations are true for proofs in MALL:

Symmetric Reductions

(7) $\quad Cut^z(\&_z^{x,y}(P_x,R_y),L_z^u(Q_u)) \;\succ\; Cut^x(P_x,Q[x/u])$

(8) $\quad Cut^z(\&_z^{x,y}(P_x,R_y),R_z^u(Q_u)) \;\succ\; Cut^x(R_x,Q[x/u])$

Commutative Reductions

(9) $\quad Cut^d(\&_z^{x,y}(P_{xd},Q_{yd}),R_d) \;=\; \&_z^{x,y}(Cut^d(P_{xd},R_d),Cut^d(Q_{yd},R_d))$

Theorem 2 (Weak Soundness) *Let $\mathcal{D}$ be a proof in MALL and let $\pi(\mathcal{D})$ be its π-calculus translation. Then we have:*

1. *Let $\mathcal{C}$ be any cut such that no &-rule occurs below $\mathcal{C}$. If $\mathcal{D}\succ_1\mathcal{D}'$ by a symmetric reduction applied to the cut $\mathcal{C}$ then $\pi(\mathcal{D}) \;\succ\; \pi(\mathcal{D}')$.*

2. *If the π-calculus admits distributivity of $\|$ and prefixes (including hiding) over $+$ and if $\mathcal{D} \succ_1 \mathcal{D}'$ by a commutative reduction then $\pi(\mathcal{D}) \equiv \pi(\mathcal{D}')$.*

Note that this theorem says the π-translation is sound only for eliminating certain *specific* cuts. This comes from the fact that $\succ$ is not a congruence with respect to $+$ (see the Introduction and Section 2). For example, with respect to the restrictions in Theorem 2, eliminating a cut in a term of the form $\&_z^{xy}(P,Q)\vec{w}z$ would involve contextual rewriting under a $+$.

We include some examples of Theorem 2 (Weak Soundness for Additives).

Example 1: The proof below has a $\&$ below a cut.

$$\dfrac{\dfrac{\vdash z : B, q'' : B^\perp}{\vdash z : B, w' : A^\perp \oplus B^\perp} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\vdash x : A, p : A^\perp \quad \vdash y : A, p : A^\perp}{\vdash w : A\&A, p : A^\perp}\&_1 \qquad \dfrac{\vdash x' : A, p' : A^\perp}{\vdash x' : A, w : A^\perp \oplus A^\perp}\oplus L}{\vdash p : A^\perp, x' : A}Cut_1}{\vdash t : A^\perp \oplus B^\perp, x' : A} \qquad \dfrac{\vdash y' : B, q' : B^\perp}{\vdash t : A^\perp \oplus B^\perp, y' : B}}{\vdash t : A^\perp \oplus B^\perp, w' : A\&B}\&_2}{\vdash z : B, t : A^\perp \oplus B^\perp}Cut_2$$

This reduces to (by a symmetric reduction to Cut_1, and axiom reduction)

$$\dfrac{\dfrac{\vdash z : B, q'' : B^\perp}{\vdash z : B, w' : A^\perp \oplus B^\perp} \qquad \dfrac{\dfrac{\overbrace{\vdash x' : A, p : A^\perp}^{Ix'p}}{\vdash t : A^\perp \oplus B^\perp, x' : A} \qquad \dfrac{\vdash y' : B, q' : B^\perp}{\vdash t : A^\perp \oplus B^\perp, y' : B}}{\vdash t : A^\perp \oplus B^\perp, w' : A\&B}}{\vdash z : B, t : A^\perp \oplus B^\perp}Cut_2$$

To the first proof corresponds the π-term (with Cut_1 and $\&_1$ in bold)

$$Ptz = Cut^{w'}(R^{q''}_{w'}(Izq''), \&^{x'y'}_{w'}(L^p_t(\mathbf{Cut}^w(\&^{\mathbf{xy}}_{\mathbf{w}}(Ixp, Iyp), L^{p'}_w(Ix'p'))), R^{q'}_t(Iy'q')))$$

The process of cut-elimination illustrated above corresponds to rewriting a subterm of Ptz as follows: the left hand branch before (and after) Cut_1 is eliminated corresponds to the following subterm reductions:

$$(\nu wuvp')\Big(\ \bar{w}\langle uv\rangle[u(x)(x(a)\bar{p}\langle a\rangle)+v(y)(y(b)\bar{p}\langle b\rangle)] \parallel w(uv)\bar{u}\langle p'\rangle(x'(c)\bar{p'}\langle c\rangle)\ \Big)$$

$$\begin{aligned}
&\succ \quad (\nu uvp')\Big(\ [u(x)(x(a)\bar{p}\langle a\rangle)+v(y)(y(b)\bar{p}\langle b\rangle)] \parallel \bar{u}\langle p'\rangle(x'(c)\bar{p'}\langle c\rangle)\ \Big)\\
&\succ \quad (\nu p')\Big(\ (p'(a)\bar{p}\langle a\rangle) \parallel (x'(c)\bar{p'}\langle c\rangle)\ \Big)\\
&\succ^* \quad x'(c)\bar{p}\langle c\rangle \quad \text{, after extruding } x'\text{, interaction, and eliminating bound } p'.\\
&\equiv \quad Ix'p \quad \text{,which is the denoted proof of } \vdash x' : A, p : A^{\perp}.
\end{aligned}$$

This illustrates Part 1 of Weak Soundness, except the condition " no $\&$-rule occurs below a cut". To understand this restriction, suppose we attempt to continue reduction of the proof, by a symmetric reduction of Cut_2, *noting that* $\&_2$ *is below a cut.* We implement this by continuing to reduce the π-term Ptz (having already reduced the inner subterm above.) The rightmost subterm of Ptz containing $R_t^{q'}(Iy'q')$ has the form $P_0 \equiv v'(y')(\nu q')t(r's')s'\langle q'\rangle(y'(d)\bar{q'}\langle d\rangle)$. But in the π-calculus, we can rewrite a term occurring in the scope of a sum only after choosing the relevant summand; here, the term P_0 is discarded by the previous steps. If we were to continue cut-elimination and apply a symmetric reduction to Cut_2, then in the associated reduction of the π-term Ptz, we eventually obtain an irreducible π-term which does not correspond to a proof, because of the disappearance of P_0. We leave the calculation to the reader.

Example 2

We illustrate Part 2 of Weak Soundness. The proof

$$\dfrac{\dfrac{\vdash x : A, p : A^{\perp} \quad \vdash y : A, p : A^{\perp}}{\vdash w : A \,\&\, A, p : A^{\perp}}\,\& \qquad \vdash p : A, q : A^{\perp}}{\vdash w : A \,\&\, A, q : A^{\perp}}\,Cut$$

becomes, after a commutative reduction,

$$\dfrac{\dfrac{\vdash x : A, p : A^{\perp} \quad \vdash p : A, q : A^{\perp}}{\vdash x : A, q : A^{\perp}}\,Cut \qquad \dfrac{\vdash y : A, p : A^{\perp} \quad \vdash p : A, q : A^{\perp}}{\vdash y : A, q : A^{\perp}}\,Cut}{\vdash w : A \,\&\, A, q : A^{\perp}}\,\&$$

and eventually becomes (after axiom reductions)

$$\frac{\vdash x : A, q : A^{\perp} \quad \vdash y : A, q : A^{\perp}}{\vdash w : A \,\&\, A, q : A^{\perp}} \;\&$$

We can exactly represent these proof transformations by the following π-term rewritings, *provided we allow distributivity of* $\|$ *over* $+$:

$$(\nu p)\Big(\;(\nu uv)\bar{w}\langle uv\rangle[u(x)x(a)\bar{p}\langle a\rangle + v(y)y(b)\bar{p}\langle b\rangle] \parallel p(c)\bar{q}\langle c\rangle\;\Big)$$

$$\begin{array}{ll} \equiv & (\nu p)\Big(\;(\nu uv)\bar{w}\langle uv\rangle[u(x)x(a)\bar{p}\langle a\rangle \parallel p(c)\bar{q}\langle c\rangle + v(y)y(b)\bar{p}\langle b\rangle \parallel p(c)\bar{q}\langle c\rangle]\;\Big) \\ \succ^{*} & (\nu uv)\bar{w}\langle uv\rangle[u(x)x(a)\bar{q}\langle a\rangle + v(y)y(b)\bar{q}\langle b\rangle] \;\; \text{(after axiom reductions).} \end{array}$$

3.3 The Abramsky Translation: the exponentials

Logical Rule	**π-translation**
$\dfrac{\vdots\, Q \atop \vdash \vec{u} : \Gamma}{\vdash \vec{u} : \;\Gamma, z :?A}\; w$	$W_z(Q)\vec{u}\,z \;=\; z(wdc)\bar{w}Q$
$\dfrac{\vdots\, Q \atop \vdash \vec{u} : \Gamma, x : A}{\vdash \vec{u} : \Gamma, z :?A}\; d$	$D_z^x(Q)\vec{u}\,z \;=\; (\nu x)z(wdc)\bar{d}\langle x\rangle Q$
$\dfrac{\vdots\, Q \atop \vdash \vec{u} : \Gamma, x :?A, y :?A}{\vdash \vec{u} : \Gamma, z :?A}\; c$	$C_z^{x,y}(Q)\vec{u}\,z \;=\; (\nu xy)(z(wdc)\bar{c}\langle xy\rangle)Q$
$\dfrac{\vdots\, Q \atop \vdash \vec{u} :?\Gamma, x : B}{\vdash \vec{u} :?\Gamma, z :!B}\; !$	$!_z^x(Q)\vec{u}\,z \;=\; !_z^x(Q\vec{u}\,x) \;=$ $(\nu wdc)(\bar{z}\langle wdc\rangle \parallel$ $w.(\|_{i=1,\ldots,n}\; u_i(wdc)\bar{w}) + d(x).Q\vec{u}x\; +$ $c(z'z'').[(\nu\vec{u'}\vec{u''})(\|_{i=1,\ldots,n}\; u_i(wdc)\bar{c}\langle u_i', u_i''\rangle$ $\parallel\, !_{z'}^x(Q\vec{u'}\,x) \;\parallel\; !_{z''}^x(Q\vec{u''}\,x))]$

For the rest of this paper, we assume the existence of the term $!_z^x(Q)\vec{u}\,z$ (as a solution to the appropriate recursion equation in the synchronous π-calculus). This deserves some discussion. One of our goals is to see how much of the structure of linear logic proofs is preserved in the translation to (synchronous) π-calculus terms. Milner's congruence $!Q \equiv Q \parallel !Q$ (or a variant such as $!Q \equiv !Q \parallel !Q$, etc.) equates terms in a manner which does not directly correspond to proof reduction. In particular, in linear logic the cut algebra for the exponentials (see below) allows duplication of $!Q$ only as a rewriting step, not as a congruence. Nonetheless, with such a rewriting interpretation of $!Q$, the soundness of the cut algebra for the exponentials (and, more generally, the proof net interpretation) follows, cf. Section 7 below.

Heuristically, in the above π-translation of the exponentials, the variables "w,d,c" represent the rules of weakening, dereliction, and contraction, resp. During communication (= cut-elimination), processes first identify themselves ("hand-shaking") as coming from one of these rules. Once these identifications establish that the communication can indeed occur, then cut-elimination is implemented through π-calculus rewriting (see the cut algebra below).

The Cut Algebra: Exponentials

The following equations are true for proofs in CLL:

Symmetric Reductions

$$Cut^z(W_z(P), !_z^x Q_{\vec{u}x}) \succ W_{\vec{u}}(P) \tag{10}$$

$$Cut^z(D_z^x(P_x), !_z^x Q_x) \succ Cut^x(P_x, Q_x) \tag{11}$$

$$\begin{aligned} &Cut^z(C_z^{xy}(P_{xy}), !_z^x(Q_{\vec{u}x})) \succ \\ &\qquad C_{\vec{u}}^{\vec{u'},\vec{u''}}(Cut^{z'}(Cut^{z''}(P[z'/x, z''/y], !_{z'}^x Q'), !_{z''}^x Q'')) \end{aligned} \tag{12}$$

where $Q' = Q[\vec{u'}/\vec{u}]$ (and similarly for Q'') and $C_{\vec{u}}^{\vec{u'},\vec{u''}}$ denotes an iterated sequence of contraction combinators.

Commutative Reductions

$$Cut^z(!_z^x Px, !^y u Q_{zy}) \quad = \quad !_u^y(Cut^z(!_z^x Px, Qzy)) \tag{13}$$

Theorem 3 (Weak Soundness) *Let $\mathcal{D}$ be a proof in CLL and let $\pi(\mathcal{D})$ be its π-calculus translation. Then we have: Let $\mathcal{C}$ be any cut such that no &- or* **!**-*rule occurs below $\mathcal{C}$. If $\mathcal{D} \succ_1 \mathcal{D}'$ by a symmetric reduction applied to the cut $\mathcal{C}$, then $\pi(\mathcal{D}) \succ_1 \pi(\mathcal{D}')$.*

There is no straightforward version of Soundness for the case of commutative reductions in CLL.

4 Translating MLL Proof Structures

We give a brief introduction to proof structures and nets, referring the reader to the literature [1, 13, 6, 12, 30, 16, 32] for more details.

The Gentzen rules for one-sided sequents for CLL are given in Section 3 above (in proof-term assignment form). In this section we consider multiplicative linear logic MLL [12, 14], whose formulas are built from $\{\otimes, \parr\}$ using linear negation $(\ )^{\perp}$ and de Morgan duality. In all that follows, we ignore the role of the units.

4.1 MLL Proof Structures

A *link* is an $m+n$-ary relation between formula occurrences, for some $m, n \geq 0$, $m + n \neq 0$. Suppose $X_1, \ldots, X_{m+n}$ are in a link: if $m > 0$, then $X_1, \ldots, X_m$ are called the *premises* of the link; if $n > 0$, then $X_{m+1}, \ldots, X_{m+n}$ are called the *conclusions* of the link. If $m = 0$, the link is called an *axiom* link. Links are graphically represented as

$$\frac{X_1, \ldots, X_m}{X_{m+1}, \ldots, X_{m+n}}$$

We consider only *logical axioms* and *multiplicative links* of the forms

$$\frac{}{A \quad A^{\perp}} \qquad \frac{A \quad A^{\perp}}{cut} \qquad \frac{A \quad B}{A \otimes B} \qquad \frac{A \quad B}{A \parr B}$$

The first two links are known as *axiom* and *cut* links, respectively. We assume that the axiom and cut links are symmetric relations. Following common practice, we shall sometimes avoid writing the word "cut" in a cut link.

A *proof structure* $\mathcal{S}$ for propositional **MLL** consists of (i) a nonempty set of *formula-occurrences* (i.e., a multiset of formulas) together with (ii) a set of logical axioms and multiplicative links satisfying the properties:

1. Every formula-occurrence in $\mathcal{S}$ is the conclusion of one and only one link;

2. Every formula-occurrence in $\mathcal{S}$ is the premise of at most one link.

We shall draw proof structures in the familiar way as non- empty, not necessarily planar, graphs.

Proof structures for **MLL** can be defined inductively, i.e., $\mathcal{S}$ is a proof structure if it results from a finite number of applications of the following clauses:

(i) an axiom $\overline{X \quad X^{\perp}}$ is a proof structure;

(ii) if $\mathcal{S}'$ and $\mathcal{S}''$ are proof structures, then so is $\mathcal{S}' \cup \mathcal{S}''$;

(iii) if $\begin{array}{c}\mathcal{S}\\ X \quad X^{\perp}\end{array}$ and $\begin{array}{c}\mathcal{S}'\\ X \quad Y\end{array}$ are proof structures, then so are

$$\begin{array}{c}\mathcal{S}\\ \underline{X \quad X^{\perp}}\end{array} \qquad \text{and} \qquad \begin{array}{c}\mathcal{S}'\\ X\ Y\\ \hline X \otimes Y\end{array} \qquad \text{and} \qquad \begin{array}{c}\mathcal{S}'\\ X\ Y\\ \hline X \parr Y\end{array}$$

We define the following reductions on proof structures:

Axiom Reductions

$$\begin{array}{ccc}\vdots & & \\ \underline{X \qquad \overline{X^{\perp}} } & \!\!\!\!\overline{\quad\ } X & \\ & \vdots & \end{array} \qquad \text{reduces to} \qquad \begin{array}{c}\vdots\\ X\\ \vdots\end{array}$$

Symmetric Reductions

$$\begin{array}{cc}\begin{array}{cc}\vdots_1 & \vdots_2\\ X & Y\\ \hline \multicolumn{2}{c}{X \otimes Y}\end{array} & \begin{array}{cc}\vdots_3 & \vdots_4\\ X^{\perp} & Y^{\perp}\\ \hline \multicolumn{2}{c}{X^{\perp} \parr Y^{\perp}}\end{array}\\ \hline \end{array}$$

reduces to

$$\begin{array}{cc} \vdots_1 & \vdots_3 \\ X & X^{\perp} \\ \hline \end{array} \qquad \begin{array}{cc} \vdots_2 & \vdots_4 \\ Y & Y^{\perp} \\ \hline \end{array}$$

Definition. A *Danos-Regnier switching* s in a proof structure consists in the choice, for each *par* link, of one of the premises of the link.

Definition. Given a proof structure $\mathcal{S}$ and a switching s, we define the undirected *Danos-Regnier graph* $\mathcal{G}_s(\mathcal{S})$ as follows:
- the vertices of $\mathcal{G}_s(\mathcal{S})$ are the formulas of $\mathcal{S}$;
- there is an edge between vertices X and Y exactly when:

(i) X and Y are the conclusions of a logical axiom or the premises of a cut link;

(ii) X is a premise and Y the conclusion of a *times* link

(iii) Y is the conclusion of a *par* link and X is the occurrence selected by the switching s.

Definition. A proof structure $\mathcal{R}$ is a *proof net* for **MLL** if for every switching s of $\mathcal{R}$, the graph $\mathcal{G}_s(\mathcal{R})$ is acyclic and connected (i.e., an undirected *tree*).

We shall denote the Danos-Regnier graph as $\mathcal{G}(\mathcal{R})$ (omitting the switching s) if there is no ambiguity, and refer to it as the *D-R graph of* $\mathcal{R}$. For further information, cf. [12, 6].

Occasionally (e.g. in Section 5.3 below) we will consider the system of sequent calculus for multiplicative linear logic, with the additional structural rule of Mix, also called *direct logic* **DL** [7, 11, 31, 4, 10]:

$$mix: \quad \frac{\vdash \Gamma \qquad \vdash \Delta}{\vdash \Gamma, \Delta}$$

Definition. A proof structure $\mathcal{R}$ is a *proof net* for *Direct Logic* **DL** if for every switching s of $\mathcal{R}$, the graph $\mathcal{G}_s(\mathcal{R})$ is acyclic (but not necessarily connected.)

The following fundamental result (Girard [13]) relates sequent calculus and proof nets for **MLL**.

Theorem 4 (Sequentialization Theorem) *There exists a map* $(\ .\ )^-$ *from sequent derivations in* **MLL** *to proof nets for* **MLL** *with the following properties:*

(a) Let $\mathcal{D}$ *be a derivation of* Γ *in the sequent calculus for* **MLL***; then* $(\mathcal{D})^-$ *is a proof net with conclusions* Γ*.*
(b) (Sequentialization) If $\mathcal{R}$ *is a proof net with conclusions* Γ *for* **MLL***, then there is a sequent calculus derivation* $\mathcal{D}$ *of* Γ *such that* $\mathcal{R} = (\mathcal{D})^-$*.*
(c) If $\mathcal{D}$ *reduces to* $\mathcal{D}'$*, then* $\mathcal{D}^-$ *reduces to* $(\mathcal{D}')^-$*.*
(d) If $\mathcal{D}^-$ *reduces to* $\mathcal{R}'$ *then there is a* $\mathcal{D}'$ *such that* $\mathcal{D}$ *reduces to* $\mathcal{D}'$ *and* $\mathcal{R}' = (\mathcal{D}')^-$*.*

A similar result can be stated for direct logic ([11, 7, 31]).

Given an MLL proof structure $\overset{\mathcal{S}}{_{A_1,\ldots,A_n}}$ with distinguished conclusions $A_1, \ldots, A_n$ we want to associate to it a basic synchronous π-calculus term $\pi(\overset{\mathcal{S}}{_{A_1,\ldots,A_n}})$ whose free names will be in bijective correspondence with the conclusions. The key point of this translation is that communication of π terms may only occur in correspondence with a cut link through a hidden channel. The hiding of this channel is explicitly given by a ν binding.This imposes a severe restriction on the behaviour of π-calculus terms in the image of our translation: free names will never communicate. This point will become clear in Proposition 4 below. As we show, the graphical properties of a proof structure will be faithfully reflected by the nesting of the bindings in the associated π-terms. Theorem 7 and its Corollaries below show this translation fully mirrors (in the π-calculus) the reduction process of proof structures (and thus proof nets). Throughout this section, we will only be using the basic synchronous π-calculus in Section 2.

4.2 Translation of MLL proof structures

The theorem below generalizes the entire discussion of Section 3.1 to the level of *structures*, emphasizing one of our main points: that the communication in the π-calculus, insofar as it relates to linear logic, is really only about the pluggings in proof structures, not the logic itself.

It is convenient to use the notation $(\nu x = y)$ defined and implemented as follows:

$$(\nu x = y)Pxy \quad =_{def} \quad (\nu zy)(\overline{z}\langle y\rangle \parallel z(x)Pxy)$$

$$\begin{array}{ll} \succ & (\nu y)Pxy[y/x] \\ \equiv & (\nu u)Pxy[u/x, u/y] \end{array}$$

We now define a map π from proof structures to terms of the synchronous π-calculus. This mapping depends upon some fixed "typing assignment" of distinct names (= variables) to all formula occurrences, which we suppose fixed once and for all.

1. $\pi(\overline{A \quad A^{\perp}}) = x(u)\,\bar{y}\langle u\rangle$, represented as

$$\overline{x : A \quad y : A^{\perp}}$$

   where x, y are in the (given) typing assignment.

2. Let $\pi(\overset{\mathcal{R}}{{}_{A\;\;B\;\;\Gamma}}) = Pxy\tilde{x}$ and let t be a name not occurring in P; then

$$\pi\left(\begin{array}{cc} \mathcal{R} & \\ \dfrac{A \quad B}{A \otimes B} & \Gamma \end{array}\right) =_{def} (\nu xy)\,\bar{t}\langle xy\rangle\,(Pxy\tilde{x})$$

   represented as

$$\begin{array}{c} P\tilde{x} \\ \dfrac{x : A \qquad\qquad y : B}{t : A \otimes B}\ \otimes \end{array}$$

3. let $\pi(\overset{\mathcal{R}}{{}_{A\quad A^{\perp}}}) = Pxy$. Then

$$\pi\left(\begin{array}{c} \mathcal{R} \\ \underline{A \quad A^{\perp}} \end{array}\right) =_{def} (\nu x = y)Pxy$$

   represented as

$$\begin{array}{c} P\tilde{x} \\ \underline{x : A \qquad\qquad x : A^{\perp}} \end{array}$$

4. Let $\pi(\overset{\mathcal{R}}{{}_{A\;\;B\;\;\Gamma}}) =_{def} Pxy\tilde{x}$ and let p be a name not occurring in P; then

$$\pi\left(\begin{array}{cc} \mathcal{R} & \\ \dfrac{A \quad B}{A \parr B} & \Gamma \end{array}\right) =_{def} p(xy)(Pxy\tilde{x})$$

represented as

$$\frac{\overset{P\tilde{x}}{x : A \qquad\qquad y : B}}{p : A \parr B}\ \parr$$

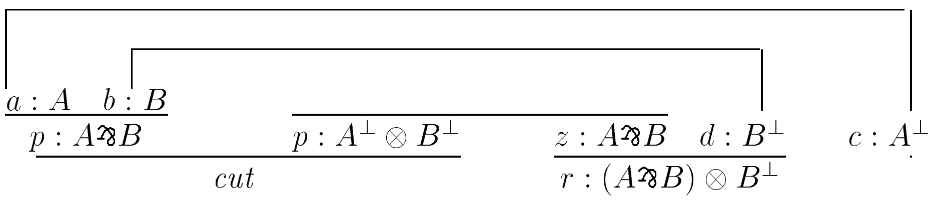
5. Let $\pi(\underset{A\ B\ \ \Gamma}{\mathcal{R}})$ and $\pi(\underset{C\ D\ \ \Delta}{\mathcal{S}})$ be the translations of two proof structures. Then

$$\pi\left(\underset{A\ B\ \ \Gamma}{\mathcal{R}} \quad \cup \quad \underset{C\ D\ \ \Delta}{\mathcal{S}}\right) \quad = \quad \pi(\underset{A\ B\ \ \Gamma}{\mathcal{R}}) \parallel \pi(\underset{C\ D\ \ \Delta}{\mathcal{S}})$$

To fix terminology, in a prefix $a(x_1, \ldots, x_n)$ or $\overline{a}\langle x_1, \ldots, x_n\rangle$ the occurrences of a or $\overline{a}$ are called *names in channel position* (or simply *channels*) and the occurrences of $x_1, \ldots, x_n$ are called *names in message position* (or simply *messages*). Among names in channel position, $\overline{a}\langle\quad\rangle$ is a *sender* and $a(\quad)$ is a *receiver*.

Remark The notation in the above translation is intended to signify the following: if we erase the variable names, we obtain the proof structure; if we erase the formulas, the notation indicates that the lowermost variables are the free names of the π term and the rest of the graph represents the binding structure of the variables in channel position.

Example The following is a proof structure $\mathcal{R}$ which is *not* a proof net:



It has the following translation into a π term:

$$(\nu z d)\overline{r}\langle z d\rangle(\nu p)(p(ab)(I_{ac} \parallel I_{bd}) \parallel I_{pz})$$

where I denotes the translation of the appropriate axiom link, e.g. I_{ac} is the term $a(u)\overline{c}\langle u\rangle$, corresponding to the axiom link $\overline{a : A \quad c : A^{\perp}}$. (Recall we only consider atomic axiom links.) We remark that, by Proposition 5 below, this translation will not depend on the order of the inductive construction of the proof structure.

Example (Deadlock) The deadlocked proof structure $\overline{A \quad A^{\perp}}$ has the translation $\nu x[x(a)\bar{x}\langle a\rangle]$.

The next proposition gives some important syntactic properties of those π-terms which translate linear logic proofs:

Proposition 5 *Let $P = \pi(\mathcal{R})$, for some proof structure $\mathcal{R}$. Then the following are true, modulo α-conversion:*

(i) Every name occurs at most once or twice in channel position.

(ii) If x occurs twice in channel position in a process P, then both occurrences are bound by the same (νx), i.e. P can be written as $(\nu x = y)Pxy$, where x is a sender and y a receiver. In particular, each free name x occurs at most once in a channel position. ("a communicating channel corresponds to a cut");

(iii) If $\overline{a}\langle x_1, \ldots, x_n\rangle$ occurs in a process P , then $x_1, \ldots, x_n$ are bound, either by some $c(x_1, \ldots, x_n)$, $(n = 1)$ or by $(\nu x_1, \ldots, x_n)$, $(n = 2)$. ("senders send only private names") Therefore, if $\bar{y}ab$ and $\bar{z}cd$ occur in a process P,then a, b, c, d are all distinct. ("different channels send different messages")

(iv) A message in a unary prefix is bound by (or binds) exactly one other unary prefix ("unary channels correspond to axiom links");

(v) If x and y occur as messages in a binary prefix, then both x and y occur in P also in channel position. ("binary channels correspond to binary links");

Since proof structures are inductively generated, it is possible to characterize exactly those π- terms which arise as the translation of proof structures, by adding some clauses to Proposition 4. We shall refrain from doing this here.

Proposition 6 *If $\mathcal{R}$ is a proof structure, then $\pi(\mathcal{R})$, modulo $\equiv$, does not depend on the particular order of the inductive construction of $\mathcal{R}$.*

Proof. By induction on the number of links in $\mathcal{R}$. E.g., let $\mathcal{R}$ be

$$\begin{array}{ccc} & \mathcal{R}'' & \\ \dfrac{B \quad C}{B \otimes C} & & \dfrac{B^\perp \quad C^\perp}{B^\perp ⅋ C^\perp} \end{array}$$

and assume the lemma for $\pi(\mathcal{R}'') = P''xyuv$. Then in the synchronous π-calculus we certainly have

$$p(uv)\,(\nu xy)\,\bar{t}\langle xy\rangle\,P''xyuv \equiv (\nu xy)\,\bar{t}\langle xy\rangle\,p(uv)\,P''xyuv$$

■

We now state two theorems which completely characterize the relationship between proof structure reducibility (and thus for proof nets) and the π-calculus translation considered here.

Theorem 7 (Weak Soundness) *If $\mathcal{R}, \mathcal{S}$ are proof structures for MLL, and if $\mathcal{R} \succ \mathcal{S}$ then $\pi(\mathcal{R}) \succ \pi(\mathcal{S})$.*

Proof: This is a straightforward induction on the formation of proof structures. ■

The above result generalizes to proof structures Abramsky's result for proofs stated earlier.

Theorem 8 (Local Fullness) *For any proof structure $\mathcal{R}$, if $\pi(\mathcal{R}) \succ Q$, then there is a proof structure $\mathcal{S}$ such that $Q = \pi(\mathcal{S})$ and $\mathcal{R} \succ \mathcal{S}$, as illustrated* *by the following diagram:*

$$\begin{tikzcd}
\mathcal{R} \arrow[d, mapsto, "\pi"'] & \succ & \mathcal{S} \arrow[d, dashed, "\pi"] \\
\pi(\mathcal{R}) & \succ & Q
\end{tikzcd}$$

Proof. Suppose $\pi(\mathcal{R}) \succ Q$. Since $\mathcal{R}$ is inductively generated, so is $\pi(\mathcal{R})$ and we can argue by induction on the formation of $\pi(\mathcal{R})$. Then by Proposition 5, part (i), (ii) and (iii), communication can occur only along a hidden channel

$$(*)\qquad \begin{aligned} &\pi_{x_1,\ldots,x_n}(\nu a)\,(\overline{a}\langle x_1,\ldots,x_n\rangle P_{x_1,\ldots,x_n} \parallel a(y_1,\ldots,y_n)Q_{y_1,\ldots,y_n}) \\ &\qquad \succ \pi_{x_1,\ldots,x_n}\,(P_{x_1,\ldots,x_n} \parallel Q_{y_1,\ldots,y_n}[x_1/y_1,\ldots,x_n/y_n]) \end{aligned}$$

Here $Q_{y_1,\ldots,y_n}$ is the scope of the binder $a(y_1,\ldots,y_n)$ and $\pi_{x_1,\ldots,x_n}$ is a prefix binding $x_1,\ldots,x_n$ whose scope in $\pi(\mathcal{R})$ does not extend beyond the subterm indicated in $(*)$.

If $n = 1$, then by part (iv) Q_{y_1} is a sender $\overline{b}\langle y_1\rangle$ and communication $(*)$ corresponds to a reduction

$$\begin{array}{ccc} \vdots & & \\ \underline{a : X^{\perp} \quad \overline{a : X} } & \!\!\!\!\overline{\quad\quad b : X^{\perp}} & \\ cut & \vdots & \end{array} \qquad \text{reduces to} \qquad \begin{array}{c} \vdots \\ b : X^{\perp} \\ \vdots \end{array}$$

i.e., the desired $\mathcal{S}$ is obtained from $\mathcal{R}$ by eliminating the cut and an axiom link as indicated.

If $n = 2$, then by part (v) communication $(*)$ corresponds to a reduction from

$$\dfrac{\dfrac{\begin{array}{c}\vdots_1\\ x_1 : X\end{array} \qquad \begin{array}{c}\vdots_2\\ x_2 : Y\end{array}}{a : X \otimes Y} \qquad \dfrac{\begin{array}{c}\vdots_3\\ y_1 : X^{\perp}\end{array} \qquad \begin{array}{c}\vdots_4\\ y_2 : Y^{\perp}\end{array}}{a : X^{\perp} \parr Y^{\perp}}}{cut}$$

to

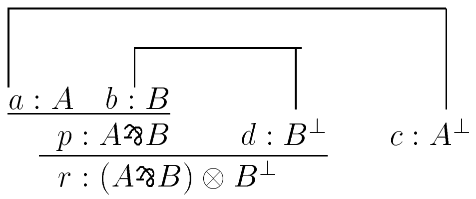
$$\dfrac{\begin{array}{c}\vdots_1\\ x_1 : X\end{array} \quad \begin{array}{c}\vdots_3\\ x_1 : X^{\perp}\end{array}}{cut} \qquad\qquad \dfrac{\begin{array}{c}\vdots_2\\ x_2 : Y\end{array} \quad \begin{array}{c}\vdots_4\\ x_2 : Y^{\perp}\end{array}}{cut}$$

i.e., the desired $\mathcal{S}$ is obtained from $\mathcal{R}$ by replacing the *times*, *par* and *cut* links with two *cuts* as indicated. ■

Example (of Theorem 7): The proof structure $\mathcal{R}$ above Proposition 5 was translated by the term

$$\pi(\mathcal{R}) \;=\; (\nu zd)\overline{r}\langle zd\rangle(\nu p)(p(ab)(I_{ac} \parallel I_{bd}) \parallel I_{pz})$$

It is easy to verify that $\pi(\mathcal{R}) \;\succ\; (\nu zd)\overline{r}\langle zd\rangle z(ab)(I_{ac} \parallel I_{bd})$. Call this latter term Q. But then Q is the π-calculus translation $Q \equiv \pi(\mathcal{S})$ of the following structure $\mathcal{S}$ (and moreover $\mathcal{R} \succ \mathcal{S}$):



Corollary. *If $\pi(\mathcal{R}) \succ \pi(\mathcal{S})$ and $\mathcal{R}$ is a proof net, then $\mathcal{S}$ is a proof net too.*

An alternate proof of local fullness for MLL proof structures will be considered in Section 5.3.

5 Information Flow

The reader may have observed that there are some arbitrary choices made in the above π-calculus translation: (i) we have chosen axiom buffers to be unidirectional– atoms become receivers, and negations of atoms senders, (ii) the *times* and *par* links become *senders* and *receivers* respectively. This has a very serious consequence: the translation is not preserved under substitution for propositional variables, since it depends on the identification of an atomic occurrence in an axiom. For this reason in the original Abramsky translation, axiom buffers were *bidirectional*, i.e. Abramsky translated the axiom $\vdash\ x\ :\ A, y : A^{\perp}$ as

$$\pi(\overline{x : A \quad y : A^{\perp}}) \;=\; x(a)\bar{y}\langle a\rangle + y(a)\bar{x}\langle a\rangle$$

However, the use of $+$ creates other problems, which will be discussed later in Section 6.1. Instead, we shall here develop an *intrinsic* notion of information flow for pure nets, thus obviating the need for bidirectional buffers [3].

Terms of the λ-calculus have a natural *direction*, namely from the inputs to the output. This is obvious from the intended functional interpretation, as well as the dynamics of evaluation. V. Danos, L. Regnier [11, 30, 22], J. van de Wiele and others have studied *pure nets*, a formalism of 'proof nets' for untyped λ-calculus: pure nets are nets built from "formulas" I and O (representing input and output, resp.), using links of the forms

$$\overline{I \quad O} \qquad \frac{O \quad I}{cut} \qquad \frac{?I \quad O}{O}\,\wp \qquad \frac{!O \quad I}{I}\,\otimes \qquad \frac{O}{!O}\,! \qquad \frac{?I \ldots ?I}{?I}\,contr$$

One motivation for this formalism is to think of the $\wp$-link as satisfying the domain "equation" $D \cong\, !D \multimap D$ (where we let D be O). This is just the familiar domain equation $D \cong D \Rightarrow D$, under the Girard translation into linear logic [13]. Other notions of pure nets for untyped lambda calculus arise by imposing different domain equations [21].

One of the goals of the above work on pure nets is to understand information flow in the process of β-reduction. But the same questions arise in the typed setting. When we consider natural deduction derivations in intuitionistic logic under the Curry-Howard correspondence (i.e. qua simply typed lambda terms), we find the same flow of information in the process of normalization, and a direction in the derivations statically considered (the *elimination part* of the proof-tree becomes the *input part* and the *introduction part* of the proof-tree becomes the *output part*).

Proof nets could be designed as input-output graphs as well but, because of De Morgan dualities for classical (linear) connectives and contraposition laws, one expects that the roles of input and output should be fully interchangeable. In what follows we make this precise. Following Bellin and Van de Wiele [8], we show that (at least for MLL) we can always assign input-output directions to proof nets so that one arbitrarily chosen conclusion is the output and all the others the inputs; when this is done, we have an interpretation of the proof net as a natural deduction derivation. Conversely, all natural deduction derivations correspond to a proof net with an

[3]The problem of substitution for propositional variables relates to whether the proofs-as-processes view extends to second-order linear logic. We leave that issue open.

input-output orientation. Each translation of a classical net into intuitionistic natural deduction obviously yields a linear λ term, under the Curry Howard correspondence.

Communication between π-calculus processes also has a direction. Agents pass *names*, i.e., access to information; some agents are *senders*, others are *receivers*. A process of transfer of information, when regarded as a whole, has a certain direction. Its implementation in the π-calculus may involve a sequence of intermediate interactions – some of which may simply be identification protocols, where the information flows in both directions. These may have no direct logical meaning. Nevertheless, it makes sense to ask that an efficient π-calculus translation of some logical system should fundamentally reflect the flow of information in the 'object calculus'. In section 5.5 below we examine in more detail Milner's direct translation of linear λ-calculus into the π-calculus, and briefly compare it to the one arising from the proofs-as-processes (and pure net) viewpoint.

5.1 Pure structures

It is convenient to prove Soundness and Local Fullness in a more abstract form, which does not depend on the particulars of the above translation. For this purpose, we introduce the notion of an untyped *pure structure*, in analogy to the notion of *pure net* used in the study of untyped lambda calculi [22, 11, 30]

Definition. A *pure structure* $\mathcal{P}$ is a proof structure built from occurrences of the symbols I, O, (for *input* and *output*) together with a set of *links* of the form

$$\frac{}{I \quad O} \qquad \frac{M \quad N}{cut} \qquad \frac{M_1 \quad M_2}{N}$$

where M, M_i and N are occurrences of the symbols I, O.

Remarks

(1) Notice that in pure structures we do *not* necessarily require premisses of cuts to be complementary: for example, $\frac{I \quad I}{cut}$ represents a deadlocked pure structure.

(2) Pure nets are a special class of pure structures in our sense (with additional links).

(3) For the purpose of studying the translation of multiplicative proof structures, the above links suffice. A more general setting (possibly useful for studying larger classes of π-calculus terms) might allow axiom links of the forms $\overline{O\ O_1\ \ldots\ O_p}\qquad \overline{I\ O_1\ \ldots\ O_p}$, with $p \geq 0$.

Pure structures, just like ordinary proof structures, can also be defined inductively: cf. section 4 above. Also in analogy with ordinary proof structures, we define the following *reductions* on pure structures:

1. *Axiom reductions.*

$$\begin{array}{ccc} \vdots & & \\ \underline{O \quad I} \!\!\!\!\!\! & \!\!\!\!\!\! \overline{\quad O} & \\ & \vdots & \end{array} \qquad \text{reduces to} \qquad \begin{array}{c} \vdots \\ O \\ \vdots \end{array}$$

2. *Symmetric reduction:*

$$\begin{array}{c} \begin{array}{cccc} \vdots_1 & \vdots_2 & \vdots_3 & \vdots_4 \end{array} \\ \underline{M \quad N} \qquad \underline{M' \quad N'} \\ \underline{\quad I \qquad\qquad\qquad O \quad} \end{array}$$

reduces to

$$\begin{array}{cc} \begin{array}{cc} \vdots_1 & \vdots_3 \\ \multicolumn{2}{c}{\underline{M \quad M'}} \end{array} & \begin{array}{cc} \vdots_2 & \vdots_4 \\ \multicolumn{2}{c}{\underline{N \quad N'}} \end{array} \end{array}$$

Definitions. Let $\mathcal{P}$ be a pure structure and $\mathcal{S}$ be an ordinary proof structure. An injective map $\tau : \mathcal{P} \to \mathcal{S}$ is said to *preserve a link* $\mathcal{L}$ of $\mathcal{P}$ if

$$\mathcal{L}\ \frac{X_1, \ldots, X_m}{Y_1, \ldots, Yn} \mapsto \frac{\tau(X_1), \ldots, \tau(X_m)}{\tau(Y_1), \ldots, \tau(Y_n)}.$$

Similarly, an injective $\delta : \mathcal{S} \to \mathcal{P}$ preserves $\mathcal{L}$ in $\mathcal{S}$ if

$$\mathcal{L}\ \frac{X_1, \ldots, X_m}{Y_1, \ldots, Yn} \mapsto \frac{\delta(X_1), \ldots, \delta(X_m)}{\delta(Y_1), \ldots, \delta(Y_n)}.$$

An injective link-preserving map $\tau : \mathcal{P} \to \mathcal{S}$ is called a *typing* of $\mathcal{P}$; an injective link-preserving $\delta : \mathcal{S} \to \mathcal{P}$ is called an *orientation* of $\mathcal{S}$.

5.2 Translation of pure structures into the π-calculus

Let $\mathbf{P}$ be the set of pure structures and let $\mathbf{\Pi_0}$ be a set of π-calculus terms satisfying conditions $(i) - (v)$ of Proposition 4.

We define a representation π_{pure} of pure structures into π-calculus terms as follows. For each $\mathcal{P} \in \mathbf{P}$ we define a π-calculus term $\pi_{pure}(\mathcal{P})$ by induction on the definition of $\mathcal{P}$. We will let $\pi_{pure}(I)$ be a receiver $x(\ )$, and $\pi_{pure}(O)$ a sender $\overline{x}\langle\ \rangle$. To verify that $\pi_{pure}(\mathcal{P}) \in \Pi_0$, in particular, that all free names in $\pi_{pure}(\mathcal{P})$ are pairwise distinct, we make sure that at each step of the construction "fresh" free names are used. This can be implemented as follows. We have two lists ℓ_f and ℓ_b of names; at the beginning of the process of translation we have an infinite ℓ_f and empty ℓ_b. Let concatenation be denoted by "."

(i) $\pi_{pure}(\overline{I\quad O}) = a(x)\overline{b}\langle x\rangle)$, where $\ell_f = a.b.\mathrm{rest}\ell_f$; now set $\ell_f := \mathrm{rest}\ell_f$, and $\ell_b := x.\ell_b$ respectively;

(ii) if $\mathcal{P} = \mathcal{P}' \cup \mathcal{P}''$, then $\pi_{pure}(\mathcal{P}) = \pi_{pure}(\mathcal{P}') \parallel \pi_{pure}(\mathcal{P}'')$, where $\pi_{pure}(\mathcal{P}')$ and $\pi_{pure}(\mathcal{P}'')$ are previously given.

(iii) if $\pi_{pure}({}_{\mathcal{M}}{}^{\mathcal{P}}{}_{\mathcal{N}}) = P_{x,y}$, then

$$\pi_{pure}\left(\begin{array}{c}\mathcal{P}\\ \underline{M\quad N}\end{array}\right) \quad = \quad (\nu x = y)P_{x,y} = (\nu z)P_{x,y}[z/x, z/y]$$

where $z = \mathrm{first}(\ell_f)$. Now set $\ell_f := \mathrm{rest}\ell_f$ and $\ell_b := z.x.y.\ell_b$.

(iv) if $\pi_{pure}({}_{\mathcal{M}'}{}^{\mathcal{P}}{}_{\mathcal{M}''}) = P_{x,y}$, then

$$\pi_{pure}\left(\begin{array}{c}\mathcal{P}\\ M\quad N\\ \hline I\end{array}\right) = a(x,y)P_{x,y}$$

and

$$\pi_{pure}\left(\begin{array}{c} \mathcal{P} \\ M \quad N \\ \hline O \end{array}\right) = (\nu x, y)\overline{a}\langle x, y\rangle P_{x,y}$$

where $a = \text{first}(\ell_f)$. Now set $\ell_f := \text{rest}(\ell_f)$ and $\ell_b := x.y.\ell_b$.

Remark: There is also a dual translation $\pi^{\perp}_{pure}$ which interchanges the rôles of sender and receiver, i.e. the interpretation of I and O.

Analogously to the previous Theorem 6 (Soundness), one obtains for pure structures:

Theorem 9 (Soundness) *If $\mathcal{P}, \mathcal{P}'$ are pure structures for MLL such that $\mathcal{P} \succ \mathcal{P}'$, then $\pi_{pure}(\mathcal{P}) \succ \pi_{pure}(\mathcal{P}')$ and also $\pi^{\perp}_{pure}(\mathcal{P}) \succ \pi^{\perp}_{pure}(\mathcal{P}')$.*

We now prove local fullness for pure structures:

Theorem 10 (Local Fullness) *For any pure structure $\mathcal{P}$, if $\pi_{pure}(\mathcal{P}) \succ Q$, then there is a pure structure $\mathcal{P}'$ such that $Q = \pi_{pure}(\mathcal{P}')$ and $\mathcal{P} \succ \mathcal{P}'$, as illustrated by the following diagram:*

$$\begin{tikzcd} \mathcal{P} \arrow[d, mapsto, "\pi_{pure}"'] & \succ & \mathcal{P}' \arrow[d, dashed, "\pi_{pure}"] \\ \pi_{pure}(\mathcal{P}) & \succ & Q \end{tikzcd}$$

and similarly for $\pi^{\perp}_{pure}$

Proof The proof is the same as that given in section 4.2., with 'pure structure' in place of 'proof structures'. ∎

5.3 Deadlock-free Structures

The next useful result shows that the two π-calculus translations of pure structures and ordinary proof structures (respectively) are the same, modulo an *orientation*:

Proposition *Let $\mathcal{S}$ be a proof structure for* **MLL** *with atomic axioms only. Let $\mathcal{P}$ be the pure structure with the same form as $\mathcal{S}$ but with all formula occurrences replaced by I's and O's as follows: atoms map to I, negations of atoms map to O . Then there is a unique orientation $\delta : \mathcal{S} \to \mathcal{P}$ which maps atoms to I, negations of atoms to O, satisfying*

$$(\S\S) \qquad \overline{\delta(X)\ \delta(X^{\perp})} = \overline{I \quad O}, \qquad \delta(X \otimes Y) = O, \qquad \delta(X \parr Y) = I.$$

and which makes the following diagram commute:

$$\begin{tikzcd} \mathcal{S} \arrow[r, dotted, "\delta"] \arrow[dr, "\pi"'] & \mathcal{P} \arrow[d, "\pi_{pure}"] \\ & \pi(\mathcal{S}) \end{tikzcd}$$

The proof is immediate.

Proposition. *Let $\mathcal{S}$ be a proof structure for* **MLL** *with atomic axioms only and let δ be an orientation satisfying (§§).*

(i) *If $\mathcal{S} \succ \mathcal{S}'$, then $\delta(\mathcal{S}) \succ \delta(\mathcal{S}')$.*

(ii) *If $\delta(\mathcal{S}) \succ \mathcal{P}'$, then we can find a $\mathcal{S}'$ such that $\mathcal{S} \succ \mathcal{S}'$ and $\mathcal{P}' = \delta(\mathcal{S}')$.*

Proof. A reducible cut link $\underline{P \quad P^{\perp}}$ or $\underline{A \parr B \quad A^{\perp} \otimes B^{\perp}}$ in $\mathcal{S}$ is mapped to a cut link of the form $\underline{I \quad O}$, which is reducible. Conversely, a reducible cut link $\underline{I \quad O}$ in $\delta(\mathcal{S})$ can only be the image of a reducible cut $\underline{P \quad P^{\perp}}$ or $\underline{A \parr B \quad A^{\perp} \otimes B^{\perp}}$ in $\mathcal{S}$. ∎

We now give an alternate proof of Local Fullness:

Theorem 11 (Local Fullness) *For any MLL proof structure $\mathcal{S}$, if $\pi(\mathcal{S}) \succ Q$ as π-calculus terms, then there is a proof structure $\mathcal{S}'$ such that $Q = \pi(\mathcal{S}')$ and $\mathcal{S} \succ \mathcal{S}'$, as illustrated by the following diagram:*

$$\begin{tikzcd}
\mathcal{S} \arrow[r, phantom, "\succ"] \arrow[d, mapsto, "\pi"'] & \mathcal{S}' \arrow[d, dashed, "\pi"] \\
\pi(S) \arrow[r, phantom, "\succ"] & Q
\end{tikzcd}$$

Proof: Consider the following diagram:

$$\begin{tikzcd}
\mathcal{S} \arrow[rr, "\succ"] \arrow[dr, "\delta"] \arrow[dd, "\pi"'] & & \mathcal{S}' \arrow[d, dotted, "\delta"] \\
 & \mathcal{P} \arrow[r, "\succ"] \arrow[dl, "\pi_{pure}"] & \mathcal{P}' \arrow[d, "\pi_{pure}"] \\
\pi(\mathcal{S}) \arrow[rr, "\succ"] & & \pi(\mathcal{P}')
\end{tikzcd}$$

By the local fullness of pure structures, there is a pure structure $\mathcal{P}'$ filling-in the bottom cell of the diagram. We thus must find a proof structure $\mathcal{S}'$ and an orientation δ with the above properties. The previous two Propositions complete the proof. ■

Definitions.

(i) A pure structure $\mathcal{P}$ is called *deadlock-free* if no links of the form $\underline{I \quad I}$, $\underline{O \quad O}$ or $\overline{I \quad O}$ occur in $\mathcal{P}$.

(ii) A reduction sequence $\mathcal{P}_0 \succ \mathcal{P}_1 \succ \ldots \succ \mathcal{P}_n$ is deadlock-free if every $\mathcal{P}_i$ is deadlock free.

(iii) An orientation $\delta : \mathcal{S} \to \mathcal{P}$ is *computationally consistent* if every reduction sequence starting from $\mathcal{S}$ is mapped by δ to a deadlock-free reduction sequence.

What are the general structures that are guaranteed to be deadlock- free?

We recall (see Section 4.1) that a proof structure satisfying the acyclicity condition (but not necessarily connectedness) is a proof net for *Direct Logic* **DL** namely, multiplicative linear logic **MLL** with the structural rule of *Mix* [6, 4, 16, 10].

Theorem 12 (Deadlock-free nets) *Let $\mathcal{S}$ be a proof net for MLL or direct logic and let δ be an orientation of $\mathcal{S}$ satisfying (§§). Then δ is computationally consistent.*

Proof. The reader should check that under the assignment (§§) to an axiom reduction of proof nets (Section 4.1) there corresponds an axiom reduction of pure structures (Section 5.1) and to a symmetric reduction of proof nets there corresponds a symmetric reduction of pure structures where M is the dual of M' and N is the dual of N'. It remains to show that the orientation $\overline{\underline{IO}}$ cannot be obtained as a result of the cut-elimination process, i.e., that the configuration $\overline{\underline{XX^{\perp}}}$ cannot be obtained as a result of the cut-elimination process for proof-nets. This follows from the well-known fact that cut-elimination preserves acyclicity of the D-R graphs. To check it, suppose that a proof-structure $\mathcal{R}'$ comes from $\mathcal{R}$ by a symmetric reduction and that for some switching s, $s(\mathcal{R}')$ contains a cycle. Various cases occur depending on whether or not the cycle passes through one or more of the 'new' cut links in $\mathcal{R}$; in each case one may choose a switching for the indicated *par* links which yields a cycle in $\mathcal{R}$, contrary to the hypothesis. For instance, if the cycle passes through both the 'new' cut links then either it has the form (a) $\ldots \underline{X \quad X^{\perp}} \ldots \underline{Y \quad Y^{\perp}} \ldots$ or the form (b) $\ldots \underline{X^{\perp} \quad X} \ldots \underline{Y \quad Y^{\perp}} \ldots$. In case (a) in $\mathcal{R}$ take a Left switch for the new *par* link; in (b), $\ldots X \ldots Y, X \otimes Y, X \ldots$ is a cycle in $s(\mathcal{R})$ for any choice of the switch for the new *par* link, etc. ∎

Corollary. *Let $\mathcal{S}_1$, $\mathcal{S}_2$ be proof nets for direct logic, let $\mathcal{S}$ result from $\mathcal{S}_1$ and $\mathcal{S}_2$ by adding a cut link between them. If δ_1, δ_2 are orientations on $\mathcal{S}_1$, $\mathcal{S}_2$,* *respectively, satisfying* (§§)*, then the induced orientation* $\delta = \delta_1 \cup \delta_2$ *on* $\mathcal{S}$ *is computationally consistent.* ∎

Remark. The converse statement, that every computationally consistent pure structure is typable in direct or linear logic, is false. Indeed there are incorrect proof-structures that reduce to a proof-net; for instance

$$\begin{array}{c} \overline{A^{\perp}\quad A}\qquad\overline{A\qquad A^{\perp}}\quad\overline{A^{\perp}\qquad A} \\ \quad A\otimes A \qquad\quad A^{\perp}\parr A^{\perp} \qquad \\ \hline \mathit{cut} \end{array} \quad \text{reduces to} \quad \begin{array}{c} \overline{A^{\perp}\quad A}\qquad\overline{A^{\perp}\quad A}\qquad\overline{A^{\perp}\quad A} \\ \qquad\mathit{cut}\qquad\qquad\mathit{cut}\qquad \end{array}$$

Therefore the following pure structure is computationally consistent

$$\begin{array}{c} \overline{I\qquad O}\quad\overline{O\qquad I}\quad\overline{I\qquad O} \\ \quad O \qquad\qquad I \qquad\qquad \\ \hline \mathit{cut} \end{array} \quad \text{reduces to} \quad \begin{array}{c} \overline{I\qquad O}\quad\overline{I\qquad O}\quad\overline{I\qquad O} \\ \qquad\mathit{cut}\qquad\qquad\mathit{cut}\qquad \end{array}$$

So this pure structure, when typed, does not become a proof-net for either linear or direct logic. Therefore our notion of computational consistency, which arises naturally from the consideration of cut-elimination as logical computation and from the notion of deadlock for a π-calculus term, is not sufficient to characterize proof-nets for direct logic.

Intuitively, the acyclicity condition for D-R-graphs (which characterizes proof-nets in direct logic) seems to express absence of deadlock in a logical computation; clearly, a stronger notion of deadlock than ours is needed to characterize correctness for such nets. We shall not pursue the matter further here (cf. recent work of A. Asperti [5]).

Another example of computationally consistent orientation will be considered in Section 5.4 below.

5.4 Orienting Proof Nets and Linear λ-terms

The present section is based on joint work by G.Bellin and J. van de Wiele ([8]).

The system **IMLL** of intuitionistic multiplicative linear logic is based on a language with the connectives $\multimap$ and $\otimes$; there is a symbol for *falsity* (we use $\perp$ as in classical linear logic) but no property of falsity is assumed for here. Linear negation is defined as $A^{\perp} =_{df} A \multimap \perp$. The sequent calculus rules and the associated term assignment (of "linear" lambda terms) are familiar [20, 1]. For example, in addition to λ-abstraction and substitution, the following rules for $\otimes$ are assumed:

$$\otimes R: \ \frac{\vec{y} : \Gamma \vdash t : A \qquad \vec{z} : \Delta \vdash u : B}{\vec{y} : \Gamma, \vec{z} : \Delta \vdash t \otimes u : A \otimes B}$$

$$\otimes L: \ \frac{\vec{y} : \Gamma, x : A, y : B \vdash t : C}{\vec{y} : \Gamma, z : A \otimes B \vdash \texttt{let } z \texttt{ be } x \otimes y \texttt{ in } t : C}$$

For simplicity, we consider only proof nets $\mathcal{S}$ for **MLL** with the property that *every axiom link contains an atomic formula and its linear negation.* Given such a proof net $\mathcal{S}$, consider orientations $\delta : \mathcal{S} \to \{O, I\}$ (which can also be written as $\delta : \mathcal{S} \to \mathcal{P}$) satisfying the following restrictions:

$$\text{axiom: } \overline{O \quad I} \qquad\qquad \text{cut: } \underline{I \quad O}$$

$$\text{tensor: } (1)\frac{O \quad I}{I} \qquad (2)\frac{I \quad O}{I} \qquad (3)\frac{O \quad O}{O}$$

$$\text{par: } (4)\frac{I \quad O}{O} \qquad (5)\frac{O \quad I}{O} \qquad (6)\frac{I \quad I}{I}$$

As usual[13], axiom and cut links are assumed symmetric in I and O.

Theorem 13 (Bellin, van de Weile) *The following hold:*

(a) If $\mathcal{S}$ is a proof net with a chosen conclusion A, and s is a D-R switching for $\mathcal{S}$, then s determines an orientation $\delta_s : \mathcal{S} \to \{I, O\}$ satisfying $(1) - (6)$*. Moreover, if the chosen conclusion is assigned O, all the other conclusions are assigned I.*

(b) There exists a map $(\ .\ )^i$ from **IMLL** *sequent derivations to* **MLL** *proof nets with orientations, satisfying:*

(b_1) *Let $\mathcal{D}$ be a derivation of $\Gamma \vdash A$ in the sequent calculus for* **IMLL***; then $(\mathcal{D})^i$ is a proof net $\mathcal{S}$ with an orientation $\delta : \mathcal{S} \to \{I, O\}$; the conclusion A^δ of $\mathcal{S}$ is equivalent to A in classical* **MLL**.

(b_2) *If an orientation δ for $\mathcal{S}$ is computationally consistent, then there exists a formula A^δ and derivation $\mathcal{D}$ of A^δ in the sequent calculus* **IMLL** *such that $(\mathcal{D})^i = \mathcal{S}$.*

(b_3) *If $\mathcal{S}$ is a proof-net with cut and s' is a partial switching undefined above all cut-links, then there exists a total switching s such that the orientation δ determined by s is computationally consistent.*

(b_4) *If $\mathcal{D}$ reduces to $\mathcal{D}'$, then $\mathcal{D}^i$ reduces to $(\mathcal{D}')^i$.*

(b_5) *If $(\mathcal{S}, \delta) = \mathcal{D}^i$ and $\mathcal{S}$ reduces to $\mathcal{S}'$ then there is a $\mathcal{D}'$ such that $\mathcal{D}$ reduces to $\mathcal{D}'$ and $\mathcal{S}' = (\mathcal{D}')^i$.*

Using the linear λ-term assignment to **IMLL** sequents, one obtains from (b):

Corollary 14 *Let M be a linear λ-term as above. Then M corresponds to a proof net with orientation $\delta : \mathcal{S} \to \{I, O\}$. Conversely, if $\mathcal{S}$ is a proof net with conclusion C, then every computationally consistent orientation $\delta : \mathcal{S} \to \{I, O\}$ determines a linear λ-term of type C^δ.* ∎

Thus, an oriented proof net for classical **MLL** yields a linear λ-term, whose output type is assigned O and whose arguments are assigned I.

Proof of Theorem. (a): Let $\mathcal{S}$ be a proof net with conclusions $C_1, \ldots, C_n$, and let s be a D-R switching for $\mathcal{S}$ together with a choice of a conclusion C_i. We define an orientation $\delta : \mathcal{S} \to \{I, O\}$ as follows.

- Let $C_i \mapsto O$. Starting from C_i, we follow the D-R graph $s(\mathcal{S})$.
- Proceeding *upwards* – from the conclusion to both premises of a *times* link and from the conclusion of a *par* link to the premise which is chosen by the switch – we always assign O to formulae encountered *(O-paths)*;
- When, proceeding upwards, we reach an axiom $\overline{P \quad Q}$ and assign O, say, to P, then we continue by letting $Q \mapsto I$ and proceed downwards from Q;

- proceeding *downwards* – from a premise to the conclusion of a *times* link and from the selected premise to the conclusion of a *par* link – we always assign I for formulas encountered *(I-paths)*; however

- In the case of a *times* link $\dfrac{A_0 \quad A_1}{A_0 \otimes A_1}$ if $A_i \mapsto I$ (and hence $A_0 \otimes A_1 \mapsto I$), then we let $A_{1-i} \mapsto O$ and also start another process proceeding *upwards* from A_{1-i}; the case of a *cut* link is similar; and so on.

Each subroutine of the procedure ends either with a premise of a *par* link which is not selected by the switch for that link or at a conclusion, in both cases after having assigned I to the formula-occurrence in question. Since the D-R graph is acyclic, the $I - O$ assignment is unique and the procedure terminates; since the D-R graph is connected, the assignment is total. Thus we have an orientation $\delta : \mathcal{S} \to \{I, O\}$ such that if $\delta(C_i) = O$, then $\delta(C_j) = I$, for all $j \neq i$.

Remark. It can also be proved that any $I - O$ assignment satisfying conditions (1)-(6) above, when applied to any proof net yields $C_i \mapsto O$ for some conclusion of the proof-net in question. (In particular, this holds when applied to a subnet of an oriented proof net.) Indeed, given $\delta : \mathcal{S} \to \{I, O\}$ and a conclusion C_j with $\delta(C_j) = I$, we proceed as follows:

- proceeding *upwards*, follow the unique *I-path* up to an axiom;

- proceeding *downwards* from an axiom, follow the unique *O-path* until either (i) a conclusion C_i is reached with $\delta(C_i) = O$, or (ii) a premise A_i of a *times* link such that $\delta(A_i) = O$ but for the conclusion we have $\delta(A_0 \otimes A_1) = I$;

- since in case (ii) above we must have $\delta(A_{1-i}) = I$, we continue upwards from A_{1-i}, and so on.

Since the D-R graph is acyclic, case (b) can occur only a finite number of times; thus eventually we reach a conclusion C_i of $\mathcal{S}$ such that $\delta(C_i) = O$, as claimed (end of remark).

Proofs of (b), (b_1)-(b_3) (the rest is straightforward): Girard's Sequentialization Theorem (see section 4.1) provides a map $(\ .\ )^{-}$ from classical **MLL** sequent proofs (thus from **IMLL** proofs) to proof nets (cf. also [20]) which determines a bijection between the inferences of a derivation $\mathcal{D}$ and the links of $\mathcal{S} = (\mathcal{D})^-$. The active formulas and the principal formula of an inference in $\mathcal{D}$ correspond to the premises and the conclusion, respectively, of a link in $\mathcal{S}$. Orientations exist, by (a). This provides the map $(\ .\ )^i$ in (b).

Case I: Cut free proof nets. We consider first the translation of *cut-free* proof nets (which are trivially computationally consistent). Given an orientation $\delta : \mathcal{S} \to \{I, O\}$, by Girard's theorem let $\mathcal{D}$ be a *classical* sequent derivation such that $\mathcal{S} = (\mathcal{D})^-$. We simultaneously define the *intuitionistic* derivation $\mathcal{D}^\delta$ and for each formula A in $\mathcal{D}^\delta$, the translation A^δ. We write A_O if $\delta(A) = O$ and A_I if $\delta(A) = I$; we write A^δ_O for the translation of A when $\delta(A) = O$, etc.

- For P atomic, we let $P^\delta_O = P = (P^\perp_I)^\delta$ and $P^\delta_I = P^\perp = (P^\perp_O)^\delta$. Thus

$$\vdash P^\perp_I, P_O \quad \mapsto \quad P \vdash P \qquad\qquad \vdash P^\perp_O, P_I \quad \mapsto \quad P^\perp \vdash P^\perp$$

- Given A^δ_O, B^δ_O, let $(A \otimes B)^\delta_O = A^\delta_O \otimes B^\delta_O$; thus

$$\frac{\vdash \Gamma_I, A_O \qquad \vdash \Delta_I, B_O}{\Gamma_I, \Delta_I \vdash (A \otimes B)_O} \quad \mapsto \quad \frac{\Gamma^\delta_I \vdash A^\delta_O \qquad \Delta^\delta_I \vdash B^\delta_O}{\Gamma^\delta_I, \Delta^\delta_I \vdash A^\delta_O \otimes B^\delta_O}$$

- Given A^δ_O, B^δ_I, let $(A \otimes B)^\delta_I = A^\delta_O \multimap B^\delta_I$; thus

$$\frac{\vdash \Gamma_I, A_O \qquad \vdash B_I, \Delta_I, C_O}{\vdash \Gamma_I, (A \otimes B)_I, \Delta_I, C_O} \quad \mapsto \quad \frac{\Gamma^\delta_I \vdash A^\delta_O \qquad B^\delta_I, \Delta^\delta_I \vdash C^\delta_O}{\Gamma^\delta_I, A^\delta_O \multimap B^\delta_I, \Delta^\delta_I \vdash C^\delta_O}$$

- Conversely, given A^δ_I and B^δ_O, let $(A \otimes B)^\delta_I = B^\delta_O \multimap A^\delta_I$ and proceed symmetrically to the previous case.

- Given A^δ_I, B^δ_O, let $(A \parr B)^\delta_O = A^\delta_I \multimap B^\delta_O$; thus

$$\frac{\vdash \Gamma_I, A_I, B_O}{\vdash \Gamma_I, (A \parr B)_O} \quad \mapsto \quad \frac{\Gamma^\delta_I, A^\delta_I \vdash B^\delta_O}{\Gamma^\delta_I \vdash A^\delta_I \multimap B^\delta_O}$$

- Conversely, given A^δ_O and B^δ_I, let $(A \parr B)^\delta_O = B^\delta_I \multimap A^\delta_O$ and proceed symmetrically to the previous case.

- Finally, given A^δ_I, B^δ_I, let $(A \parr B)^\delta_I = A^\delta_I \otimes B^\delta_O$; thus

$$\frac{\vdash \Gamma_I, A_I, B_I, C_O}{\vdash \Gamma_I, (A \parr B)_I, C_O} \quad \mapsto \quad \frac{\Gamma^\delta_I, A^\delta_I, B^\delta_I \vdash C^\delta_O}{\Gamma^\delta_I, A^\delta_I \otimes B^\delta_I \vdash C^\delta_O}$$

Case II: proof nets with Cut. In the case of a proof-net $\mathcal{S}$ with Cut links, by the sequentialization theorem we obtain a sequent derivation $\mathcal{D}$ containing subderivations

$$\dfrac{\begin{array}{c}\mathcal{D}'\\ \vdots\\ \vdash \Gamma, A\end{array} \quad \begin{array}{c}\mathcal{D}''\\ \vdots\\ \vdash \Delta, A^{\perp}\end{array}}{\vdash \Gamma, \Delta}$$

and we need to show that $A_I^\delta = (A^\perp)_O^\delta$ or $A_O^\delta = (A^\perp)_I^\delta$. This is not true in general, if we take arbitrary switchings on the *par* links occurring above A and $A^\perp$ in $\mathcal{S}$. Let $\mathcal{R}'$ be the subnet of $\mathcal{S}$ given by $(\mathcal{D}')^-$ and similarly, let $\mathcal{R}'' = (\mathcal{D}'')^-$. Consider the pure structures $\delta(\mathcal{R}') = \mathcal{P}'$ and $\delta(\mathcal{R}'') = \mathcal{P}''$; in $\mathcal{P}'$ and $\mathcal{P}''$ consider the subtrees corresponding to the hereditary subformulas of A and $A^\perp$: what we need is that such subtrees be *dual*, e.g.,

$$\dfrac{\dfrac{\dfrac{I \quad O}{I} \quad \dfrac{I \quad O}{O}}{O} \quad \dfrac{\dfrac{O \quad I}{O} \quad \dfrac{O \quad I}{I}}{I}}{cut}$$

Using the cut-elimination theorem we show that this requirement can be met, by induction on the length of the cut-elimination procedure.

• If the proof net $\mathcal{R}_1$ reduces to $\mathcal{R}_2$ by an *axiom reduction*

$$\begin{array}{c}\vdots\\ \underline{P \qquad \overline{P^{\perp} \qquad P}}\\ \qquad\qquad\ \vdots\end{array} \qquad \text{reduces to} \qquad \begin{array}{c}\vdots\\ P\\ \vdots\end{array}$$

and $\delta_2 : \mathcal{R}_2 \to \{I, O\}$ is computationally consistent, then it is immediate to define a computationally consistent $\delta_1 : \mathcal{R}_1 \to \{I, O\}$, as no new *par* links are to be considered.

• Suppose $\mathcal{R}_1$ reduces to $\mathcal{R}_2$ by a *symmetric reduction*

$$\dfrac{\dfrac{\begin{array}{c}\vdots_1\\ A\end{array} \quad \begin{array}{c}\vdots_2\\ B\end{array}}{A \otimes B} \qquad \dfrac{\begin{array}{c}\vdots_3\\ A^\perp\end{array} \quad \begin{array}{c}\vdots_4\\ B^\perp\end{array}}{A^\perp ⅋ B^\perp}\,\mathcal{L}}{}$$

reduces to

$$\begin{array}{cc} \vdots_1 & \vdots_3 \\ \hline A & A^{\perp} \end{array} \qquad \begin{array}{cc} \vdots_2 & \vdots_4 \\ \hline B & B^{\perp} \end{array}$$

and suppose we have a computationally consistent orientation $\delta_2 : \mathcal{R}_2 \to \{I, O\}$ so that $A^\delta = (A^\perp)^\delta$ and $B^\delta = (B^\perp)^\delta$: we need to choose a switching for the new *par* link $\mathcal{L}$ so as to extend δ_2 to a computationally consistent $\delta_1 : \mathcal{R}_1 \to \{I, O\}$. By Girard's sequentialization theorem the above reduction step corresponds to the reduction from $\mathcal{D}_1$ to $\mathcal{D}_2$, say, by the steps

$$\text{from} \quad \dfrac{\dfrac{\begin{array}{c}\mathcal{D}'\\ \vdots \\ \vdash \Gamma, A\end{array} \quad \begin{array}{c}\mathcal{D}''\\ \vdots \\ \vdash \Delta, B\end{array}}{\vdash \Gamma, \Delta, A \otimes B} \quad \dfrac{\begin{array}{c}\mathcal{D}'''\\ \vdots \\ \vdash \Lambda, A^\perp, B^\perp\end{array}}{\vdash \Lambda, A^\perp ⅋ B^\perp}}{\vdash \Gamma, \Delta, \Lambda} \qquad \text{to} \quad \dfrac{\begin{array}{c}\mathcal{D}'\\ \vdots \\ \vdash \Gamma, A\end{array} \quad \dfrac{\begin{array}{c}\mathcal{D}''\\ \vdots \\ \vdash \Delta, B\end{array} \quad \begin{array}{c}\mathcal{D}'''\\ \vdots \\ \vdash \Lambda, A^\perp, B^\perp\end{array}}{\vdash \Delta, \Lambda, A^\perp}}{\vdash \Gamma, \Delta, \Lambda}$$

Notice that $\mathcal{R}''' = (\mathcal{D}''')^-$ is a subnet of $\mathcal{R}_2 = (\mathcal{D}_2)^-$, and that by part (a) *at most one* of $A^\perp$, $B^\perp$ can be assigned value O by δ_2.

- *case* (i) $\delta_2(A^\perp) = O$: let the switch for the link $\mathcal{L}$ be Left;
- *case* (ii) $\delta_2(B^\perp) = O$: let the switch for the link $\mathcal{L}$ be Right;
- *case* (iii) $\delta_2(A^\perp) = I = \delta_2(B^\perp)$: choose the switch for $\mathcal{L}$ arbitrarily.

In case (ii) we obtain δ_1 extending δ_2 with

$$\begin{array}{c} \begin{array}{cc} \vdots_1 & \vdots_2 \\ \hline O & I \end{array} \qquad \begin{array}{cc} \vdots_3 & \vdots_4 \\ \hline I & O \end{array} \\ \hline I \qquad\qquad\quad O \end{array}$$

which reduces to

$$\begin{array}{cc} \vdots_1 & \vdots_3 \\ \hline O & I \end{array} \qquad \begin{array}{cc} \vdots_2 & \vdots_4 \\ \hline I & O \end{array}$$

Letting $\Delta = \Theta, C$, we obtain a translation thus:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Gamma_I^{\delta_1} \vdash A_O^{\delta_1}\end{array} \quad \begin{array}{c}\vdots\\ B_I^{\delta_1}, \Theta_I^{\delta_1} \vdash C_O^{\delta_1}\end{array}}{\Gamma_I^{\delta_1}, A_O^{\delta_1} \multimap B_I^{\delta_1}, \Theta_I^{\delta_1} \vdash C_O^{\delta_1}} \quad \dfrac{\begin{array}{c}\vdots\\ \Lambda_I^{\delta_1}, (A^\perp)_I^{\delta_1} \vdash (B^\perp)_O^{\delta_1}\end{array}}{\Lambda_I^{\delta_1} \vdash (A^\perp)_I^{\delta_1} \multimap (B^\perp)_O^{\delta_1}}}{\Gamma_I^{\delta_1}, \Theta_I^{\delta_1}, \Lambda_I^{\delta_1} \vdash C_O^{\delta_1}}$$

Indeed using the induction hypothesis we obtain

$$B_I^{\delta_1} = B_I^{\delta_2} = (B^\perp)_O^{\delta_2} = (B^\perp)_O^{\delta_1} \qquad \text{and} \qquad (A^\perp)_I^{\delta_1} = (A^\perp)_I^{\delta_2} = A_O^{\delta_2} = A_O^{\delta_1}$$

hence $A_O^{\delta_1} \multimap B_I^{\delta_1} = (A^\perp)_I^{\delta_1} \multimap (B^\perp)_I^{\delta_1}$ and the indicated Cut is correct.

Case (i) is symmetric.

In case (iii) we obtain δ_1 extending δ_2 with

$$\dfrac{\dfrac{\overset{\vdots_1}{O} \quad \overset{\vdots_2}{O}}{I} \qquad \dfrac{\overset{\vdots_3}{I} \quad \overset{\vdots_4}{I}}{O}}{}$$

which reduces to

$$\dfrac{\overset{\vdots_1}{O} \quad \overset{\vdots_3}{I}}{} \qquad \dfrac{\overset{\vdots_2}{O} \quad \overset{\vdots_4}{I}}{}$$

Letting $\Lambda = \Xi, D$, we obtain a translation thus:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Gamma_I^{\delta_1} \vdash A_O^{\delta_1}\end{array} \quad \begin{array}{c}\vdots\\ \Delta_I^{\delta_1} \vdash B_O^{\delta_1}\end{array}}{\Gamma_I^{\delta_1}, \Delta_I^{\delta_1} \vdash A_O^{\delta_1} \otimes B_O^{\delta_1}} \quad \dfrac{\begin{array}{c}\vdots\\ (A^\perp)_I^{\delta_1}, (B^\perp)_I^{\delta_1}, \Xi_I^{\delta_1} \vdash D_O^{\delta_1}\end{array}}{(A^\perp)_I^{\delta_1} \otimes (B^\perp)_O^{\delta_1}, \Xi_I^{\delta_1} \vdash D_I^{\delta_1}}}{\Gamma_I^{\delta_1}, \Delta_I^{\delta_1}, \Xi_I^{\delta_1} \vdash D_O^{\delta_1}}$$

indeed, using the induction hypothesis we obtain

$$A_O^{\delta_1} = A_O^{\delta_2} = (A^\perp)_I^{\delta_2} = (A^\perp)_I^{\delta_1} \qquad \text{and} \qquad B_O^{\delta_1} = B_O^{\delta_2} = (B^\perp)_I^{\delta_2} = (B^\perp)_I^{\delta_1}$$

hence $A_O^{\delta_1} \otimes B_O^{\delta_1} = (A^\perp)_I^{\delta_1} \otimes (B^\perp)_I^{\delta_1}$.

Finally it is not difficult to show that all translations are classically equivalent: we argue by induction on the length of the sequent calculus derivation and use the provable classical equivalence of $A \otimes B$ with $(A \multimap B^\perp)^\perp$ and $(B \multimap A^\perp)^\perp$, etc. ■

Remarks

1. For the purposes of the Corollary, we could argue directly and translate each proof net for **MLL** with a computationally consistent orientation $\delta : \mathcal{S} \to \{I, O\}$ into a natural deduction derivation in **IMLL**. This requires developing properties of the subnets of a proof net. Instead we prefer to use Girard's sequentialization theorem.
2. As pointed out to us by M. Hyland, the procedure for constructing orientations in (a) can be formulated as a finite game, where the opponent begins and chooses D-R switches and O-paths, while the player replies by choosing I-paths. This appears related to recent work of Hyland and Ong on game-theoretic semantics for linear logic.
3. The procedure to construct orientations in (a) can be extended to the whole system of Linear Logic, but the translation $(\ .\ )^\delta$ in (b_1)-(b_3) *cannot* be extended in general. We omit the counterexamples; however the problem is Contraction (either explicitly in the exponentials or implicitly in the side doors of additive boxes.) A reasonable theory of computationally consistent orientations for the additives has been developed, but we omit it here for reasons of space.

Example:

The proof net

$$\frac{A \qquad B}{A \otimes B} \qquad \frac{B^\perp \qquad C}{B^\perp \otimes C} \qquad \frac{C^\perp \qquad D}{C^\perp \otimes D} \qquad \frac{D^\perp \qquad A^\perp}{D^\perp \parr A^\perp}$$

with the choice of $A \otimes B$ as output receives the orientation

$$\frac{O \qquad O}{O} \qquad \frac{I \qquad O}{I} \qquad \frac{I \qquad O}{I} \qquad \frac{I \qquad I}{I}$$

These data represent the derivation

$$\dfrac{D \otimes A \qquad \dfrac{[A] \qquad \dfrac{C \multimap B \qquad \dfrac{D \multimap C \quad [D]}{C}\multimap E}{B}\multimap E}{A \otimes B}\otimes I}{A \otimes B}\otimes E$$

The following gives another orientation of the same net, with the choice of $D^\perp \parr A^\perp$ as output:

$$\dfrac{I \qquad O}{I} \qquad \dfrac{I \qquad O}{I} \qquad \dfrac{I \qquad\quad O}{I} \qquad \dfrac{I \qquad O}{O}$$

These data represent the derivation

$$\dfrac{\dfrac{B \multimap \neg A \qquad \dfrac{C \multimap B \qquad \dfrac{D \multimap C \quad [D]}{C}\multimap E}{B}\multimap E}{\neg A}\multimap E}{D \multimap \neg A}\multimap I$$

and so on. ■

Define a map γ on the hereditary premises of the cut formulas of $\mathcal{S}$ as follows. For every cut link $\mathcal{L} : \underline{X \quad X^\perp}$, let

- $X \mapsto X^\perp$
- if $\frac{Y_1 \quad Y_2}{Y}$ and $\frac{Y_1^\perp \quad Y_2^\perp}{Y^\perp}$ are hereditary premises of X, $X^\perp$, respectively, and $Y \mapsto Y^\perp$, then $Y_1 \mapsto Y_1^\perp$ and $Y_2 \mapsto Y_2^\perp$.

Lemma. *If $\mathcal{S}$ is a proof net for* direct logic *or* multiplicative linear logic *such that only atomic formulas occur in axiom links, any orientation $\delta : \mathcal{S} \to \mathcal{P}$ is computationally consistent if and only if for every hereditary premise of a cut link in $\mathcal{P}$, the map $\varphi = \delta \circ \gamma \circ \delta^{-1}$ dualizes the orientation* (i.e., $\varphi(I) = O$ and $\varphi(O) = I$).

Sketch of proof. Because of the assumption that only atomic formulas occur in axiom links, the above definition is total on the set of the hereditary premises of all cut links. Since $\mathcal{S}$ is a proof net, a configuration $\overline{\underline{X \quad X^{\perp}}}$ will never occur as a result of cut-elimination from $\mathcal{S}$. Given any orientation $\delta : \mathcal{P} \to \mathcal{S}$, together with its inverse $\tau : \mathcal{S} \to \mathcal{P}$, we prove by induction on the length of a reduction sequence that $\delta \circ \varphi \circ \tau$ inverts I and O if and only if no pure structure in the reduction sequence contains an irreducible cut. ∎

Any Danos-Regnier switching, for example, induces such an orientation, by the work in Section 5.4. We obtain a translation of proof nets into synchronous π-calculus by considering the composite $\mathcal{R} \xrightarrow{\delta} \mathcal{P} \xrightarrow{\pi_{pure}} \pi_{pure}(\mathcal{P})$. We could also use the $\pi^{\perp}_{pure}$ translation. In all cases, soundness and local fullness hold, according to section 5.2.

5.5 Milner's linear λ-terms

The following is R. Milner's direct translation [27] of the linear λ-calculus into the π-calculus :

$$[\![\lambda x M]\!]u =_{def} u(xv)\,[\![M]\!]v$$

$$[\![MN]\!]u =_{def} (\nu v)\,([\![M]\!]v \parallel (\nu x)\overline{v}\langle xu\rangle x(w)[\![N]\!]w)$$
(where x is not free in N).

$$[\![x]\!]u =_{def} \overline{x}\langle u\rangle$$

It is interesting to compare Milner's translation of the linear λ-calculus with our "logical" translations into oriented proof nets: (i) the Abramsky translation π and (ii) the translation $\pi\circ\delta$ obtained from Bellin and de Wiele's result.

Lambda abstraction

The following (intuitionist) proof represents lambda abstraction:

$$\dfrac{\begin{matrix}\vdots \\ x : A \vdash M : B\end{matrix}}{\vdash \lambda x.M : A \multimap B} \multimap \mathrm{R} \quad .$$

Using one-sided sequents (in classical linear logic) this becomes:

$$\frac{\begin{array}{c}\vdots\, M\\ \vdash x : A^{\perp}, v : B\end{array}}{\vdash \parr(M) : A^{\perp}\parr B}\;\parr \quad ,$$

which corresponds to the proof net

$$\frac{\begin{array}{c}\vdots\, \mathcal{S}\\ A^{\perp} \quad B\end{array}}{A^{\perp}\parr B} \quad ,$$

with orientation

$$\frac{\begin{array}{c}\vdots\, \delta(\mathcal{S})\\ I \quad O\end{array}}{O}$$

Let $\ll M \gg xv$ be the Abramsky translation $\pi(\mathcal{R})$. From the translation for $\parr$, we obtain:

$$\ll \lambda x.M \gg u = u(xv) \ll M \gg xv$$

Note that this is precisely the same translation as Milner obtains. But $\delta(A^{\perp}\parr B) = O$, hence the translation $\pi \circ \delta(\mathcal{R})$ has the dual form of a sender.

Application

The following (intuitionist) proof represents application MN in natural deduction

$$\frac{\begin{array}{c}\vdots\\ M : A \multimap B\end{array} \quad \begin{array}{c}\vdots\\ N : A\end{array}}{MN : B}$$

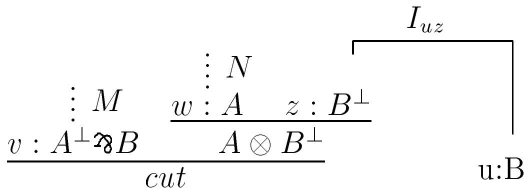
which in the sequent calculus **ILL** becomes

$$\cfrac{\cfrac{\begin{array}{c}\vdots\, M\\ \vdash A \multimap B\end{array} \quad \cfrac{A \vdash A \quad B \vdash B}{A, A \multimap B \vdash B}\,\multimap \mathrm{L}}{A \vdash B}\,\mathit{cut} \quad \begin{array}{c}\vdots\, N\\ \vdash A\end{array}}{\vdash B}\,\mathit{cut}$$

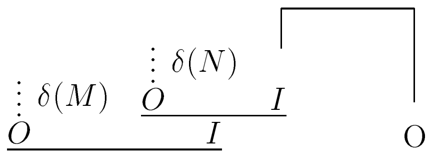
Translating into classical **LL** one-sided sequents (letting $A \multimap B$ be $A^{\perp}\parr B$) and reducing the lower-most cut yields the following representation of application

$$\cfrac{\begin{array}{c}\vdots\, M\\ \vdash A^{\perp}\parr B\end{array} \qquad \cfrac{\begin{array}{c}\vdots\, N\\ \vdash A\end{array} \qquad \vdash B^{\perp}, B}{\vdash A\otimes B^{\perp}, B}\,\otimes}{\vdash B}\,cut$$

The last proof translates into the following proof net:



with orientation



Let $\ll M \gg v$ and $\ll N \gg w$ be π-terms, with free names v (resp. w), translating the above proofs by the Abramsky translation. We have:

$$\begin{aligned} \ll MN \gg u &= \nu vwz(\ll M \gg v \parallel \bar{v}\langle wz\rangle(\ll N \gg w \parallel I_{uz})) \\ &= \nu vwz(\ll M \gg v \parallel \bar{v}\langle wz\rangle(\ll N \gg w \parallel u(b)\bar{z}\langle b\rangle)) \end{aligned}$$

We note that Milner's term for MN is essentially an "optimized" version of our term: the logic gives an additional axiom buffer I_{zu} whereas Milner's translation gives a direct implementation. We may think of this axiom buffer as a "dangling wire", which a more efficient, direct implementation would avoid.

On the other hand, in the translation $\pi\circ\delta(\mathcal{R})$, since we have $\delta(A\otimes B^{\perp}) = I$, the agent corresponding to $A \otimes B^{\perp}$ becomes a receiver, rather than a sender.

Variables

The variable x is interpreted as an axiom buffer, viz. $\overline{u : A \quad x : A^{\perp}}$. So we have:

$$\ll x \gg u = u(a)\overline{x}\langle a \rangle$$

We see that the Milner translation is more direct: instead of sending information from x to the port u via channel a, Milner directly sends u !

So how do we explain that both Milner and Abramsky translate linear λ terms using the *dual* of the $\pi{\circ}\delta$ translation and turn an O-formula into a *receiver* and an I-formula into a *sender*?

According to the basic philosophy of the π-calculus, π-terms consist of *agents* in an *interacting environment*; thus λ-calculus computations are interpreted into the π-calculus as interactions between certain agents and their environment. An environment in which a term $\lambda x.M$ can interact is one where an input can be *received* for x and a new environment is created for M with such an input. An environment v making the application MN possible can be thought as an agent *sending* information to M; in particular, such information includes the name x by which the main input channel of M is identified henceforth, if such an input exists. The name x is also given to an input channel which controls the new environment of the term N; since the names v and x are made private (i.e. are bound variables), the information flow is tightly controlled in the new process.

6 The Additives

6.1 Problems with the Additives

To handle the additive connectives of linear logic, we consider the full synchronous π- calculus. Unfortunately, the additives illustrate the divergence of pure logic from the concurrency world. We shall begin by illustrating the problems.

Problems with the Axioms

Recall, in the last section, we mentioned that the original Abramsky translation [2, 3] considers the axioms as *bidirectional buffers*. That is, one translates[4]

$$\pi(\overline{A \quad A^{\perp}}) = x(u)\,\bar{y}\langle u\rangle + y(u)\,\bar{x}\langle u\rangle$$

Consider the proof net

$$x : A \qquad\qquad \frac{y : A^{\perp} \quad y : A}{cut} \qquad\qquad z : A^{\perp}$$

Assuming axioms are bidirectional buffers, we obtain the following π-term:

$$\nu y[x(u)\overline{y}u \;+\; y(u)\overline{x}u \;\parallel y(u)\overline{z}u \;+\; z(u)\overline{y}u]$$

Allowing distributivity (of $\parallel$ with respect to $+$) in the full synchronous π-calculus causes troubles: using distributivity, we may multiply out the expressions (like polynomials). The two cross-terms $x(u)\overline{y}u \;\parallel\; z(u)\overline{y}u$ and $y(u)\overline{x}u \;\parallel\; y(u)\overline{z}u$ will be deadlocked.

There are at least three obvious choices of what to do:

1. Use unidirectional buffers, as in the last section, which must then be shown to be coherent with respect to the rest of the translation of the additives.

2. Ban distributivity at the level of translation of axioms.

3. Garbage collect deadlocked terms.

Although it seems fairly common in the concurrency world to consider garbage collection as a reasonable solution in such problems [25], the logical status of having deadlocked terms with garbage collection is somewhat unclear. It is certainly contrary to the spirit of translating logical deduction in a sound and faithful way.

Problems with Additive Reduction

[4]As will become clear below, $+$ is related to the translation of boxes. This seems to fit with the remark of Girard [13], 2.4, p.45, "Technically speaking ... axioms are boxes".

We shall now discuss some of the thorny problems with additive reductions. The reader unfamiliar with proof boxes might find it helpful to first read Section 6.2 (Proof Structures with Boxes) below, or to use it as a reference for what follows.

To analyze the representation of the additive connectives [13] in the synchronous π-calculus, consider the sequent calculus introduction rule for &

$$\frac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, A \& B}$$

where Γ is a multiset of formula occurrences (often called a *context*) .

Recall the behaviour of the cut-elimination process:

Symmetric reduction

$$\dfrac{\dfrac{\begin{matrix}\mathcal{D}_0\\ \vdots\\ \vdash \Gamma, A_0\end{matrix} \qquad \begin{matrix}\mathcal{D}_1\\ \vdots\\ \vdash \Gamma, A_1\end{matrix}}{\vdash \Gamma, A_0 \& A_1}\,\& \qquad \dfrac{\begin{matrix}\mathcal{D}_2\\ \vdots\\ \vdash A_i^{\perp}, \Delta\end{matrix}}{\vdash A_0^{\perp} \oplus A_1^{\perp}, \Delta}\,\oplus}{\vdash \Gamma, \Delta}\,cut$$

reduces to

$$\dfrac{\begin{matrix}\mathcal{D}_i\\ \vdots\\ \vdash \Gamma, A_i\end{matrix} \qquad \begin{matrix}\mathcal{D}_2\\ \vdots\\ \vdash A_i^{\perp}, \Delta\end{matrix}}{\vdash \Gamma, \Delta}\,cut$$

Commutative reduction

$$\dfrac{\dfrac{\begin{matrix}\mathcal{D}_0\\ \vdots\\ \vdash C, \Gamma, A\end{matrix} \qquad \begin{matrix}\mathcal{D}_1\\ \vdots\\ \vdash C, \Gamma, B\end{matrix}}{\vdash C, \Gamma, A \& B}\,\& \qquad \ldots}{\vdash D, \Delta, A \& B}\,\mathcal{R}$$

reduces to

$$\dfrac{\dfrac{\begin{array}{c}\mathcal{D}_0\\ \vdots\\ \vdash C, \Gamma, A \quad \ldots\end{array}}{\vdash D, \Delta, A}\ \mathcal{R} \qquad \dfrac{\begin{array}{c}\mathcal{D}_1\\ \vdots\\ \vdash C, \Gamma, B \quad \ldots\end{array}}{\vdash D, \Delta, B}\ \mathcal{R}}{\vdash D, \Delta, A\&B}\ \&$$

where $\mathcal{R}$ is a cut rule (cf. the Cut algebra in section 3.2).

In sequent calculus the introduction rule for & is *contextual* in the sense that the contexts in the premises and conclusion must be the same multiset Γ; for the inference to be correct, a test for the identity of the contexts is required.

However, proof-theory does not specify *when* such a test must be performed: by *permuting* the & rule with the rules below it (as in commutative reductions) the test could be delayed and performed on a larger context.

As pointed out by Girard [16] and Abramsky [1], the meaning of the connective & (*with*) seems to involve notions of *choice*: $A\&B$ asserts the availability of two courses of action, the choice being an "external" one, i.e. not determined by the subject who makes the assertion, hence cannot be inferred from the given deductive context. On the other hand, the dual connective $\oplus$ (*plus*) asserts an alternative "internal" choice, which *is* determined by the speaker.

The key *dynamical* point is that, with respect to symmetric reductions, the choice between the two subderivations $\mathcal{D}_0$ and $\mathcal{D}_1$ leading to $\vdash \Gamma, A\&B$ is determined during the process of cut-elimination by the information contained in the derivation $\mathcal{D}_2$ of $\vdash A^\perp \oplus B^\perp, \Delta$. If a formula C_i in $\vdash C_1, \ldots, C_n, A\&B$ becomes itself a cut formula, then the interaction expressed by the elimination of such a cut is *frozen* unless and until interaction through $A\&B$ has happened. Additive Commutative Reductions are therefore necessary if we want to either (i)change the order of communication, or (ii) to make interaction through C_i possible, if interaction through $A\&B$ never happens.

In the theory of proof-nets, the issue of finding an optimal representation for additives is still under active research. There are at least three candidates: (i) additive boxes, (ii) additive contraction links, and (iii) slices, of which (i) and (iii) find analogues in the π-calculus translations considered below.

Additive proof boxes (see Section 6.2 below) were introduced by Girard [13] to correspond to the contextually-dependent rules of sequent calculus. In particular, the requirement "testing the identity of contexts" of the &-

introduction rule is expressed by "putting in a box" the proof-nets corresponding to the derivations of the premises. As in sequent calculus, the choice of the context is not unique: additive boxes can be expanded (cf. the Additive Commutative Reductions in section 6.1 below). This representation suggests the idea of breaking the &-rule into three processes: two yielding the premises of the inference and a third one starting with the "non-logical axiom" $\vdash \Gamma, A\&B$. This seems inappropriate: we may insist that a unique flow of information runs through the three processes and therefore search for a representation of proofs without the additive box.

Additive contraction links together with a *binary &-link*

$$\frac{A \quad B}{A\&B} \qquad \frac{C_1 \quad C_1}{C_1} \qquad \cdots \qquad \frac{C_n \quad C_n}{C_n}$$

implement the idea of a unique flow of information. These links seem implicit in the &-rule of sequent calculus; but to obtain an adequate representation we must also include information corresponding to testing for the identity of contexts, i.e., that there are subnets $\mathcal{S}_1$ with conclusions A, Γ and $\mathcal{S}_2$ with conclusions B, Γ.

Slices were introduced in the Appendix of Girard's original paper [13]; the intuition seems to be that at each &-rule we really have *two* independent subarguments, one including a step $\frac{A}{A\&B}$, the other a step $\frac{B}{A\&B}$; Here additive contractions disappear, but an *external* test then verifies that the subarguments coincide from certain points of the context; i.e., from those points different slices must be identified and only the set of all slices with all these identifications has a correct logical meaning.

The different representations have different dynamical behaviours. The first two representations, in presence of the Additive Commutative Reductions, do not enjoy the the Church-Rosser property; for a simple example, consider the possible ways of eliminating the cut in the proof nets corresponding to

$$\dfrac{\dfrac{\vdash A, A^\perp \quad \vdash A, A^\perp}{\vdash A\&A, A^\perp}\,\& \qquad \dfrac{\begin{matrix}\vdots \\ \vdash A, A^\perp \otimes (B^\perp \oplus C^\perp), B\end{matrix} \quad \begin{matrix}\vdots \\ \vdash A, A^\perp \otimes (B^\perp \oplus C^\perp), C\end{matrix}}{\vdash A, A^\perp \otimes (B^\perp \oplus C^\perp), B\&C}\,\&}{\vdash A\&A, A^\perp \otimes (B^\perp \oplus C^\perp), B\&C}\,cut$$

In the representation through slices, Additive Commutative Reductions modify only the places where the external test is performed, not the slices themselves. In fact, slices were introduced by Girard [13] as an invariant in the cut-elimination process. However, if the test of correctness has to be performed during cut-elimination, the reduction steps in different slices must be correlated. Attempts to consider a mathematical theory of slices independently of the boxes of which they are slices have so far produced a rather complicated syntax.

How does our π-calculus translation relate to the additives ? The translation of the & rule given in section 3.2 corresponds closely to the translation of the corresponding proof net with additive box: indeed communication through the channel $\overline{z}$ (corresponding to the *principal door* of the box) yields a communication through one of the channels u or v, i.e., a choice of one of the summands $P_{\vec{w}\,x}$ or $Q_{\vec{w}\,y}$. The problem is that in the *synchronous* π-calculus it may also be possible to commute some of the prefixes and to obtain a communication through some channel in $\vec{w}$, which yields a completely arbitrary choice of one of the summands (cf. section 2.2). Essentially for this reason, local fullness fails for the translation of section 3.2 [5].

To recover local fullness, we reintroduce some restrictions from the *asynchronous* π calculus [25], namely the use of guarding prefixes, as in section 2.2. Since guarding prefixes prohibit rewriting in the guarded context, only certain logical reduction strategies can be represented, see theorem 14 below.

From the logical viewpoint, if would be desirable if the π-calculus could faithfully represent the categorical notion of weak product (i.e. the additive connective &, with its symmetric reductions). This would require a notion of congruence which permits rewriting under the + as well as in a guarded context, but the theory of synchronous π-calculus has not yet decided this issue.

Additive Commutative Reductions can be naturally represented in the translation of section 2.2, if distributivity of $\|$ with respect to + is permitted. This is clearly incompatible with guarding. Therefore no evaluation strategy involving additive commutative reduction can be fully represented.

To fully represent every reduction strategy we are led to avoid the use of + in Abramsky's translation, replacing it with $\|$, as first suggested in [24]. Once this idea is fully developed, it turns out to provide a translation of

[5]Categorically speaking, the Cut Algebra for the additives (symmetric reductions) expresses the fact that & must be a *weak product*; in most models & is actually a product. This is completely at odds with the behaviour of +

Girard's theory of slices, as shown in theorem 18 below.

6.2 Proof Structures with Boxes

We consider proof structures for **MLL** extended with

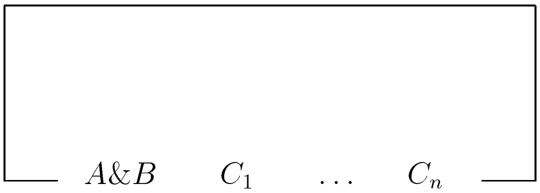
- Nonlogical Axioms i.e., links $\overline{X, X_1, \ldots, X_n}$

For Multiplicative and Additive Linear Logic **MALL** (without constants) we use the additive links

- Plus Links $\dfrac{A_0}{A_0 \oplus A_1}$ $\qquad \dfrac{A_1}{A_0 \oplus A_1}$

  and the additive boxes

- & Boxes



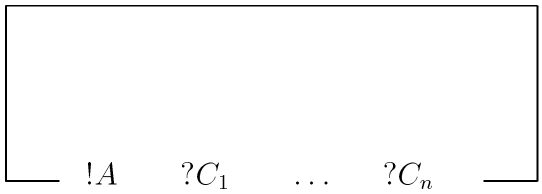
For the full propositional system (without constants) we also add links for the exponential operator ? and the **!**-*boxes* as follows:

- Weakening Axiom $\overline{?A}$

In order to verify the correctness of proof nets Weakening axioms are given with an *attachment* to some other formula occurrence (we represent it by $\overline{?A} \rightharpoondown C$).

- Dereliction Link $\dfrac{A}{?A}$
- Contraction Link $\dfrac{?A \quad ?A}{?A}$

- ! Boxes



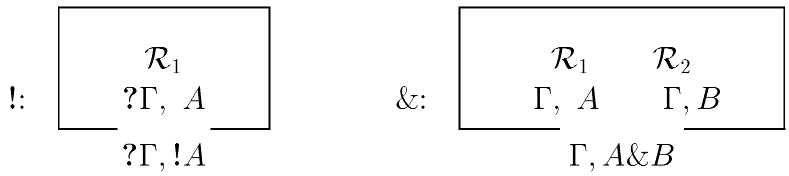
We give a precise definition of proof boxes.

Definition. A *proof box* $\mathbf{B}(\mathcal{R}_1, \ldots, \mathcal{R}_n, \mathcal{A})$ is a relation where $\mathcal{R}_1$, …, $\mathcal{R}_n$ are *pairwise disjoint* proof structures and $\mathcal{A}$ is a non-logical axiom.

Any conclusion of a proof structure $\mathcal{R}_i$ is called a *premise* of the box; occurrences in the non-logical axiom $\mathcal{A}$ are called the *conclusions* of the box $\mathbf{B}$. The premises and the conclusions of each box will be in some correspondence with each other (to be specified case by case); in all cases, a box $\mathbf{B}(\mathcal{R}_1, \ldots, \mathcal{R}_k, \overline{X, X_1, \ldots, X_n})$ has the property that for each $i \leq k$ the conclusions of $\mathcal{R}_i$ are exactly Y, X'_1, …, X'_n, where Y is an immediate (proper) subformula of X and where X'_j and X_j, $j \leq n$, are occurrences of the same formula. The occurrence X is then called the *principal door* and X_1, …, X_n the *auxiliary* doors of the box.

- In a !-box $\mathbf{B}(\mathcal{R}_1, \mathcal{A})$, the axiom $\mathcal{A}$ must have the form $\overline{!A, ?\Gamma}$; the principal door is the indicated occurrence of $!A$; the conclusions of $\mathcal{R}_1$ must be exactly $A, ?\Gamma$. (Here if $\Gamma = C_1$, …, C_n, then $?\Gamma = ?C_1$, …, $?C_n$.)
- In a &-box $\mathbf{B}(\mathcal{R}_1, \mathcal{R}_2, \mathcal{A})$, the axiom $\mathcal{A}$ must have the form $\overline{A\&B, \Gamma}$; the principal door is the indicated occurrence of $A\&B$, the conclusions of $\mathcal{R}_1$ are exactly A, Γ and the conclusions of $\mathcal{R}_2$ are exactly B, Γ.

Graphically:



In this graphical representation it is intuitively clear what it means for a formula, a substructure or a box to be *inside* or *outside* a given box.

We can define *proof structures with boxes* for full propositional Linear Logic inductively as follows. Let **lnk** be a set of correct propositional links (*logical axioms*, *weakening axioms* with attachments, *times, par, plus, dereliction, contraction* links). Then

1. a logical axiom is a proof structure with boxes;

2. if $\mathcal{R}'$ and $\mathcal{R}''$ are proof structure with boxes, the so is $\mathcal{R}' \cup \mathcal{R}''$

3. if

$$\begin{array}{c} \mathcal{R} \\ A\ A^{\perp} \end{array} \quad \text{and} \quad \begin{array}{c} \mathcal{R}' \\ A\ B \end{array} \quad \text{and} \quad \begin{array}{c} \mathcal{R}'' \\ A \end{array}$$

are proof structures with boxes and $\frac{A \quad B}{C}$ or $\frac{A}{C}$ are links in **lnk** then

$$\begin{array}{c} \mathcal{R} \\ \underline{A\ A^{\perp}} \end{array} \quad \text{and} \quad \begin{array}{c} \mathcal{R}'' \\ A \leftharpoondown ?B \end{array} \quad \text{and} \quad \begin{array}{c} \mathcal{R}' \\ \dfrac{A \quad B}{C} \end{array} \quad \text{and} \quad \begin{array}{c} \mathcal{R}'' \\ \dfrac{A}{C} \end{array}$$

are proof structures with boxes;

4. if $\mathcal{S}_1$ is a proof structure with boxes, whose conclusions are exactly $A, ?B_1, \ldots, ?B_n$ and $\mathcal{A} = \overline{!A, ?B_1, \ldots, ?B_n}$ is a nonlogical axiom, then $\mathcal{S}_1 \cup \mathcal{A}$ with the box $\mathbf{B}(\mathcal{S}_1, \mathcal{A})$ is a proof structure with boxes;

5. if $\mathcal{S}_1$ and $\mathcal{S}_2$ are proof structures with boxes, whose conclusions are exactly A, Γ and B, Γ, respectively, and if $\mathcal{A} = \overline{A\&B, \Gamma}$ is a nonlogical axiom, then $\mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{A}$ together with the box $\mathbf{B}(\mathcal{S}_1, \mathcal{S}_2, \mathcal{A})$ is a proof structure with boxes.

We may also give a direct definition of *proof structures $\mathcal{S}$ with boxes* analogous to that in Section 4.1. It is clear that if we remove the set of boxes

box from the definition of $\mathcal{S} = (\mathbf{fml}, \mathbf{lnk}, \mathbf{box})$, then we obtain a collection $(\mathbf{fml}, \mathbf{lnk}) = (\mathcal{S}_1, \ldots, \mathcal{S}_k)$ of proof structures with nonlogical axioms.

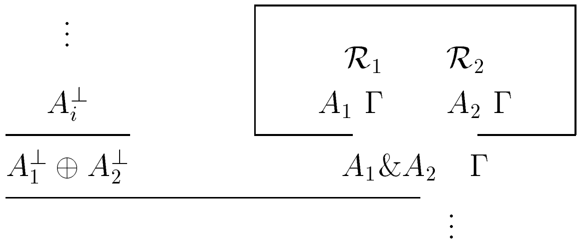
Definition. A switch for a contraction link a choice of one of the premises (cf. the *par* link). Let $\mathcal{S}$ be a proof structure with multiplicative links, and, in addition, nonlogical axioms $\overline{X, X_1, \ldots, X_n}$ and *plus*, *weakening*, *dereliction*, *contraction* links. If s is a switching for $\mathcal{S}$, then define the D-R graph as before, with the following additions:

- for each nonlogical axioms $\overline{X, X_1, \ldots, X_n}$, introduce edges (X, X_1), (X_1, X_2), ..., (X_{n-1}, X_n);
- for each *plus* link $\dfrac{A_i}{A_0 \oplus A_1}$ introduce an edge $(A_0 \oplus A_1, A_i)$; similarly for *dereliction* links;
- for each *weakening* axiom with attachment, introduce an edge corresponding to the attachment;
- for each *contraction* link, introduce an edge between the conclusion and the premise chosen by the switching (as in a *par* link).

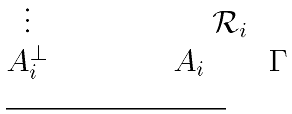
We say that $\mathcal{S}$ is a *proof net with nonlogical axioms* if for each switching s, the graph $s(\mathcal{S})$ is acyclic and connected. Finally, we say that a proof structure with boxes $\mathcal{S} = (\mathbf{fml}, \mathbf{lnk}, \mathbf{box})$ is a *proof net with boxes* if each $\mathcal{S}_i$ in $(\mathbf{fml}, \mathbf{lnk}) = (\mathcal{S}_1, \ldots, \mathcal{S}_k)$ is a proof net with nonlogical axioms.

Reductions for proof structures with additive boxes are (in addition to the previously defined MLL reductions):

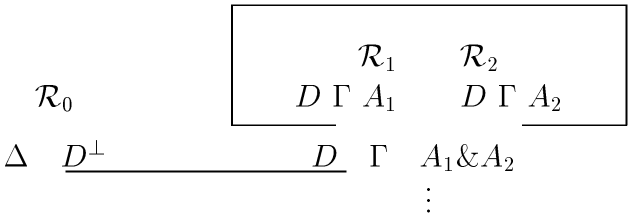
Additive Symmetric Reductions:



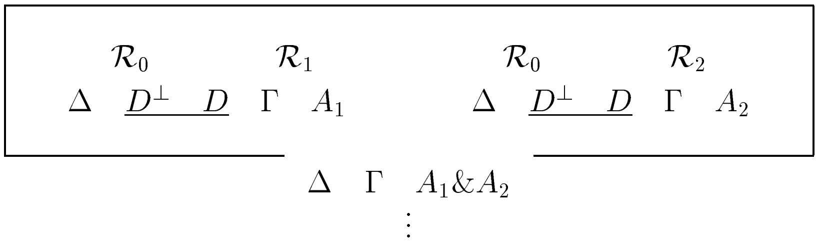
reduces to



Additive Commutative Reduction:

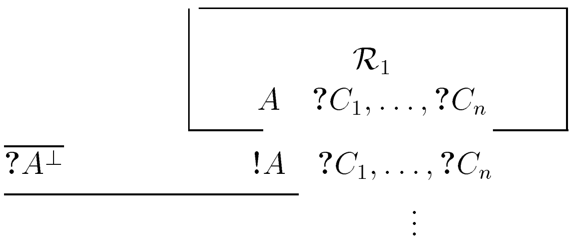


reduces to



Exponential Symmetric Reductions:

Weakening /**!**:



reduces to

$$\overline{?C_1} \ldots \overline{?C_n}$$
$$\vdots$$

Dereliction /$!$:

$$\cfrac{\cfrac{\begin{array}{c}\vdots\\ A^{\perp}\end{array}}{?A^{\perp}} \qquad \begin{array}{c}\boxed{\begin{array}{c}\mathcal{R}_1\\ A\ ?\Gamma\end{array}}\\ !A \quad ?\Gamma\end{array}}{\vdots}$$

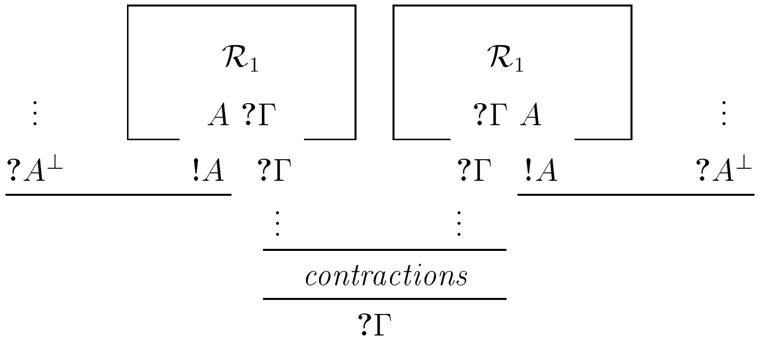
reduces to

$$\begin{array}{ccc} \vdots & & \mathcal{R}_1 \\ A^{\perp} & A & ?\Gamma \\ \hline & & \vdots \end{array}$$

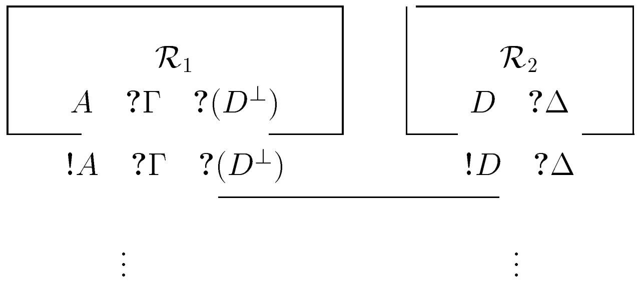
contraction /$!$:

$$\cfrac{\cfrac{\begin{array}{c}\vdots\\ ?A^{\perp} \quad ?A^{\perp}\end{array}}{?A^{\perp}} \qquad \begin{array}{c}\boxed{\begin{array}{c}\mathcal{R}_1\\ A\ ?\Gamma\end{array}}\\ !A \quad ?\Gamma\end{array}}{\vdots}$$

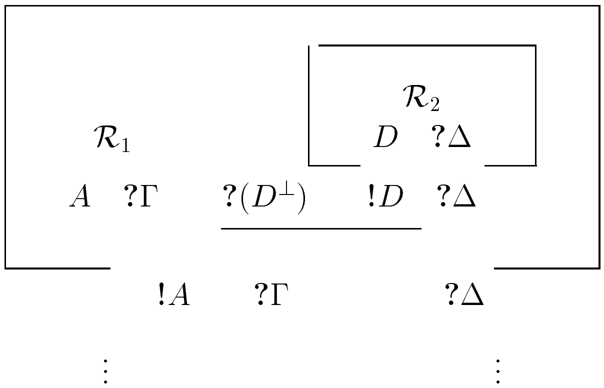
reduces to



Exponential Commutative Reductions:



reduces to



6.3 Slicings

An *additive slice* is a proof structure whose formulas are in the language of **MALL** and whose links are logical axioms, *cut*, *times* and *par* links, and in addition

$$\frac{A_i}{A_0 \oplus A_1}\oplus \qquad \text{and} \qquad \frac{A_i}{A_0 \& A_1}\& \qquad \text{for } i = 0, 1$$

A *slicing* $\mathcal{S}\ell(\mathcal{R})$ of a proof structure with boxes is a family $\langle \mathcal{Q}_i, \iota_i \rangle_{i \in I}$ where $\mathcal{Q}_i$ is a slice and $\iota_i : \mathcal{Q}_i \to \mathcal{R}$ is an embedding such that X and $\iota(X)$ are occurrences of the same formula, defined by induction on the definition of $\mathcal{R}$ as follows:

1. if $\mathcal{R}$ is a logical axiom $\mathcal{A} = \overline{X, X^{\perp}}$, then $\mathcal{S}\ell(\mathcal{R}) = \langle \mathcal{A}, id \rangle$;
2. if $\mathcal{R}$ is $\mathcal{R}_1 \cup \mathcal{R}_2$ and $\mathcal{S}\ell(\mathcal{R}_1) = \langle \mathcal{Q}_i, \iota_i \rangle_{i \in I}$ $\mathcal{S}\ell(\mathcal{R}_2) = \langle \mathcal{Q}'_j, \iota'_j \rangle_{j \in J}$ then we set $\mathcal{S}\ell(\mathcal{R}) = \langle \mathcal{Q}_i \cup \mathcal{Q}'_j, \iota_i \cup \iota'_j \rangle_{i \in I, j \in J}$;
3. let $\circ$ be either $\otimes$ or $⅋$; if

   $$\mathcal{R} = \frac{\overset{\mathcal{R}'}{A \quad B}}{A \circ B}, \qquad \text{and} \qquad \mathcal{S}\ell(\overset{\mathcal{R}'}{A \quad B}) = \langle \mathcal{Q}'_i, \iota'_i \rangle_{i \in I},$$

   then $\mathcal{S}\ell(\mathcal{R}) = \langle \mathcal{Q}_i, \iota_i \rangle_{i \in I}$ where

   $$\mathcal{Q}_i = \frac{\overset{\mathcal{Q}'}{A \quad B}}{A \circ B}$$

   and ι_i is ι'_i extended with the assignement $A \circ B \mapsto A \circ B$; similarly, if $\mathcal{R}$ results from $\mathcal{R}'$ by introducing a *cut* link, or a *plus* link;
4. let $\mathcal{A} = \overline{A_1 \& A_2, \Gamma}$; suppose $\mathcal{R}$ is $\mathbf{B}(\mathcal{R}_1, \mathcal{R}_2, \mathcal{A})$ and $\mathcal{S}\ell(\mathcal{R}_1) = \langle \mathcal{Q}'_i, \iota'_i \rangle_{i \in I}$, $\mathcal{S}\ell(\mathcal{R}_2) = \langle \mathcal{Q}''_j, \iota''_j \rangle_{j \in J}$, where I and J are disjoint; then let
   $\mathcal{S}\ell(\mathcal{R}) = \langle \mathcal{Q}_k, \iota_k \rangle_{k \in I \cup J}$, where

   $$\mathcal{Q}_k = \frac{\overset{\mathcal{Q}'_k}{A_1}}{A_1 \& A_2} \quad \text{if } k \in I, \qquad \mathcal{Q}_k = \frac{\overset{\mathcal{Q}''_k}{A_2}}{A_1 \& A_2} \quad \text{if } k \in J$$

and where ι_k is ι'_k (resp ι''_k), extended with the assignment $A\&B \mapsto A\&B$.

It is possible to consider slicings independently of any inductive definition, but we will not do this here (cf [6]).

Within each slice we define symmetric reductions as for multiplicative structures and, in addition,

$$\underline{\dfrac{\overset{\vdots}{A_i^\perp}}{A_0^\perp \oplus A_1^\perp} \qquad\qquad \dfrac{\overset{\vdots}{A_i}}{A_0\&A_1}}$$

reduces to

$$\underline{\overset{\vdots}{A_i^\perp} \qquad\qquad \overset{\vdots}{A_i}}$$

On the other hand, a slice with a link of the form

$$\underline{\dfrac{\overset{\vdots}{A_i^\perp}}{A_0^\perp \oplus A_1^\perp} \qquad\qquad \dfrac{\overset{\vdots}{A_{1-i}}}{A_0\&A_1}}$$

represents a logical inconsistency and a computational deadlock (we shall call it an *additive deadlock*) and must be deleted in the process of normalization (*garbage collection!*).

The set of slices occurring in $\mathcal{S}\ell(\mathcal{R})$ is invariant under commutative reductions (although the associated embeddings change):

Lemma. *If $\mathcal{R} \succ \mathcal{R}'$ by a commutative reduction, then*

$$\mathcal{S}\ell(\mathcal{R}) = \langle \mathcal{Q}_i, \iota_i \rangle_{i\in I} \qquad \text{and} \qquad \mathcal{S}\ell(\mathcal{R}') = \langle \mathcal{Q}_i, \iota'_i \rangle_{i\in I}$$

■

Let $\mathcal{S}$ be a proof structure with boxes. If we remove all boxes, we obtain a set $\{\mathcal{R}_1, \ldots, \mathcal{R}_n\}$ of pairwise disjoint proof structures possibly with nonlogical axioms. We say that $\mathcal{S}$ is a *proof net with boxes* if each $\mathcal{R}_i$ is a proof net with nonlogical axioms.

We now give a version of soundness and local fullness for the representation of additive proof nets (with boxes) via slicing. This is an extension of the discussion in Girard [13]; for more details see Bellin [6].

Let $\mathcal{R}$ be a proof net with boxes and consider its set of slices $\langle \mathcal{Q}_i, \iota_i \rangle_{i \in I}$. Each cut $\mathcal{C}$ in $\mathcal{R}$ corresponds to a *set* of cuts $\mathcal{C}_i$ in the set of slices (by taking the inverse images of the ι's). Reducing $\mathcal{C}$ in one step will correspond to simultaneously reducing all the $\mathcal{C}_i$. However, in the case of additive reductions, some of the $\mathcal{Q}_i$ will be in an additive deadlock. So when we do the cut-reduction, we put each deadlocked slice into a garbage collector $\mathcal{G}$ (this will be regarded as a reduction step for that slice). Thus the Soundness theorem takes the following form:

Theorem 15 (Soundness for Slicing) *Let $\mathcal{R}$ be a proof net with boxes. Let $\mathcal{R} \succ \mathcal{S}$. Then $\mathcal{S}\ell(\mathcal{R}) \succ \mathcal{S}\ell(\mathcal{S}) \cup \mathcal{G}$, where $\mathcal{G}$ is the set of slices in an additive deadlock.*

Local fullness takes the following form:

Theorem 16 (Local fullness for Slicing) *Let $\mathcal{R}$ be a proof net with boxes and let $\mathcal{C}$ be a cut therein. If $\mathcal{S}\ell(\mathcal{R}) \succ \mathcal{F}$ by a simultaneous reduction of all the cuts $\mathcal{C}_i$ in the inverse image of $\mathcal{C}$ under the ι's, then there exists a proof net $\mathcal{S}$ such that $\mathcal{R} \succ \mathcal{S}$ by reduction of the cut $\mathcal{C}$ and $\mathcal{F} = \mathcal{S}\ell(\mathcal{S}) \cup \mathcal{G}$, where $\mathcal{G}$ is a set of slices in additive deadlock.*

6.4 Translation of MALL proof structures

The π-translations given in section 3 and 4 can be extended to MALL as follows:

$$\pi\left(\begin{array}{cc} \dfrac{\overset{\mathcal{R}}{A}}{A\oplus B} & \Gamma \end{array}\right) = (\nu x)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{u}}x\,(P_{x\tilde{x}})$$

$$\pi\left(\begin{array}{cc} \dfrac{\overset{\mathcal{R}}{B}}{A\oplus B} & \Gamma \end{array}\right) = (\nu x)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{v}}x\,(P_{x\tilde{x}})$$

and

$$\pi\left(\dfrac{\boxed{\begin{array}{cc}\mathcal{R}_1 & \mathcal{R}_2\\ \Gamma,\ A & \Gamma, B\end{array}}}{\Gamma, A\&B}\right) = \nu(\mathbf{uv})\,\overline{z}\langle\mathbf{uv}\rangle.\,[\mathbf{u}(x).Q'_{x\tilde{w}} + \mathbf{v}(y).Q''_{y\tilde{w}}]$$

where $P = \pi(\mathcal{R}), Q' = \pi(\mathcal{R}_1)$, and $Q'' = \pi(\mathcal{R}_2)$.

This can be represented as:

$$\dfrac{\overset{P_{\tilde{x}}}{x : A}}{z : A\oplus B} \qquad \text{and} \qquad \dfrac{\overset{P_{\tilde{x}}}{x : A}}{z : A\oplus B}$$

and

$$\dfrac{\boxed{\begin{array}{cc} Q' & Q'' \\ x : A\ \ \tilde{w} : \Gamma & y : B\ \ \tilde{w} : \Gamma \\ \mathbf{u} & \mathbf{v} \end{array}}}{z : A\&B\ \ \tilde{w} : \Gamma,}$$

Corresponding to the additive symmetric reductions we have

$$\begin{array}{rcll}
P_0 & \equiv & (\nu z)[(\nu x)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{u}}x\, P_{x\tilde{x}} \parallel & \nu(\mathbf{u}\ \mathbf{v})\,\overline{z}\langle\mathbf{u}\ \mathbf{v}\rangle.\,[\mathbf{u}(x).Q'_{x\tilde{w}} + \mathbf{v}(y).Q''_{y\tilde{w}}]] \\
 & & & \succ \\
 & & (\nu\mathbf{u}\ \mathbf{v})\,[(\nu x)\,\overline{\mathbf{u}}x\, P_{x\tilde{x}} \parallel & [\mathbf{u}(x).Q'_{x\tilde{w}} + \mathbf{v}(y).Q''_{y\tilde{w}}]] \\
 & & & \succ \\
 & & & (\nu x)\,[P_{x,\tilde{x}} \parallel Q'_{x\tilde{w}}]
\end{array}$$

(**) and

$$
\begin{array}{rcll}
P_0 & \equiv & (\nu z)[(\nu y)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{v}}y\, P_{y\tilde{y}} \parallel & \nu(\mathbf{u}\ \mathbf{v})\, \overline{z}\langle \mathbf{u}\ \mathbf{v}\rangle .\, [\mathbf{u}(x).Q'_{x\tilde{w}} + \mathbf{v}(y).Q''_{y\tilde{w}}]] \\
 & & & \succ \\
 & & (\nu \mathbf{u}\ \mathbf{v})\, [(\nu y)\, \overline{\mathbf{v}}y\, P_{y\tilde{y}} \parallel & [\mathbf{u}(x).Q'_{x\tilde{w}} + \mathbf{v}(y).Q''_{y\tilde{w}}]] \\
 & & & \succ \\
 & & & (\nu y)\, [P_{y,\tilde{y}} \parallel Q''_{y\tilde{w}}]
\end{array}
$$

The names $\mathbf{u}$ and $\mathbf{v}$ in $(\nu x)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{u}}x\, (P_{x\tilde{x}})$ or $(\nu x)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{v}}x\, (P_{x\tilde{x}})$ or $\nu(\mathbf{uv})\, \overline{z}\langle \mathbf{uv}\rangle .\, [\mathbf{u}(x).Q'_{x\tilde{w}} + \mathbf{v}(y).Q''_{y\tilde{w}}]$ will be called *choice names*; these names determine which subprocess will be retained or discharded in the above reduction. To emphasize their rôle, we shall denote choice names by *boldface* letters in what follows, and similarly for the exponentials in the next section.

Now we prove a restricted form of soundness and local fullness of the π-translation of proof structures for MALL.

Theorem 17 *The π-translation of proof structures in* **MALL** *is sound and locally full with respect to the normalization strategy reducing the cuts which lie outside all additive boxes.*

Proof: Soundness is easy. For the local fullness, we argue by induction on the formation of $\pi(\mathcal{S})$. The only successful communications will have the forms (*) (see Theorem 7) and (**) above. The case of reduction (*) is as before, for the multiplicatives. For the sake of simplicity, we regard the two steps in (**) as a single rewriting step. Suppose that the P_0 in (**) is a subterm of P^*. By the properties of the asynchronous prefixes no interaction involving a channel in $Q'_{x\tilde{w}}$ or $Q''_{y\tilde{w}}$ may occur in P^*. Hence only interaction with the choice terms $\mathbf{u}$ or $\mathbf{v}$ can determine the choice of summand in P_0. Therefore terms of the form

$$(\nu \mathbf{u}\ \mathbf{v})\, [(\nu y)\, \overline{\mathbf{v}}y\, P_{y\tilde{y}} \parallel [\mathbf{u}(x).Q'_{x\tilde{w}}]$$

(*additive deadlock*) will never result in the reduction process. ■

Thus we see that the "communication protocols" of the choice names determine the behaviour of the relevant processes and guarantee that deadlocked slices are deleted.

We also consider a π-translation of *slicings* of a proof structure for **MALL**; to this end, we extend the translation in section 4.2. Let $\mathcal{R}$ be a proof structure with additive boxes and let $P = \pi(\mathcal{R})$. Moreover, let σ be a function defined on the set all the subterms of the form $Q_1 + Q_2$ in P which chooses either Q_1 or Q_2; let P_σ be the result of the following operations: (i) erase all guardings (i.e. dots), (ii) replace each subterm of the form Q_1+Q_2 in P with $\sigma(Q_1+Q_2)$, Call such a P_σ a *slice* of the term P. Finally, if $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ is the set of all distinct slices of P, then we define the *slicing of* P to be

$$\|_{\sigma\in\Sigma}\, P_\sigma$$

where we assume all the P_σ have disjoint free variables (if necessary, by renaming).

Alternatively, we can define the π-translation of the slices of a proof structure directly. We proceed as in the case of additive proof structures, except that for the &-links we omit guarding and take:

$$\pi\left(\begin{array}{cc} \overset{\mathcal{R}}{A} & \\ \hline A\&B & \Gamma \end{array}\right) = (\nu x \mathbf{u}\ \mathbf{v})\, \overline{z}\langle \mathbf{u}\ \mathbf{v}\rangle\, \mathbf{u}(x)\, (P_{x\tilde{x}})$$

$$\pi\left(\begin{array}{cc} \overset{\mathcal{R}}{B} & \\ \hline A\&B & \Gamma \end{array}\right) = (\nu x \mathbf{u}\ \mathbf{v})\, \overline{z}\langle \mathbf{u}\ \mathbf{v}\rangle\, \mathbf{v}(x)\, (P_{x\tilde{x}})$$

The two definitions coincide:

Proposition. *Let* $\mathcal{F} = \{\mathcal{Q}_i\}_{i\in I}$ *be the set of slices in* $\mathcal{S}\ell(\mathcal{R})$*, and let* $\Pi = \|_{\sigma\in\Sigma}\ P_\sigma$ *be the slicing of the term* $\pi(\mathcal{R})$*; then* $\pi(\mathcal{F}) = \Pi$.

In other words, there is a natural bijection between the different notions of *slices*, whether at the level of proof structures or represented as π-terms.

The following proposition is also clear:

Proposition *Let* $\mathcal{R}$ *be a proof structure with additive boxes and let* $P = \pi(\mathcal{R})$*; Proposition 5 holds for each slice of* P*, with the exception of* (iv), (v), *which fail for choice names only.* ∎

The bijection between slices of nets vs slices of π-terms extends to the operational (normalization) behaviour:

Theorem 18 *The π-translation of slicings in* **MALL** *is sound and locally full with respect to every normalization strategy.*

Proof: Since $\pi(\mathcal{S}\ell(\mathcal{R}))$ contains no $+$, it behaves as a term satisfying Proposition 5, except perhaps for choice names. These choice names can only interact in reductions of the form

$$\begin{array}{rcll} P_0 & \equiv & (\nu z)[(\nu x)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{u}}x\, P_{x\tilde{x}} \parallel & \nu(\mathbf{u}\ \mathbf{v})\, \overline{z}\langle \mathbf{u}\ \mathbf{v}\rangle.\, [\mathbf{u}(x).Q'_{x\tilde{w}}]] \\ & & & \succ \\ & & (\nu \mathbf{u}\ \mathbf{v})\, [(\nu x)\, \overline{\mathbf{u}}x\, P_{x\tilde{x}} \parallel & [\mathbf{u}(x).Q'_{x\tilde{w}}]] \\ & & & \succ \\ & & & (\nu x)\, [P_{x,\tilde{x}} \parallel Q'_{x\tilde{w}}] \end{array}$$

$(***)$ and

$$\begin{array}{rcll} P_0 & \equiv & (\nu z)[(\nu y)\, z(\mathbf{u}\ \mathbf{v})\, \overline{\mathbf{v}}y\, P_{y\tilde{y}} \parallel & \nu(\mathbf{u}\ \mathbf{v})\, \overline{z}\langle \mathbf{u}\ \mathbf{v}\rangle.\, [\mathbf{u}(x).Q'_{x\tilde{w}}]] \\ & & & \succ \\ & & (\nu \mathbf{u}\ \mathbf{v})\, [(\nu y)\, \overline{\mathbf{v}}y\, P_{y\tilde{y}} \parallel & [\mathbf{u}(x).Q'_{x\tilde{w}}]] \\ & & & \text{additive deadlock} \end{array}$$

Dually, for the slicing which deletes the left summand. Thus we see that a reduction in $\pi(\mathcal{S}\ell(\mathcal{R}))$ yields an additive deadlock if and only if so does a reduction in $\mathcal{S}\ell(\mathcal{R})$. ∎

Finally, we can extend the Bellin-van de Wiele orientations to the case of proof nets for **MALL**, as well as for the exponentials, as mentioned in the remarks after Theorem 13. Alas, *not* all such orientations are computationally consistent. The π-translations resulting from such orientations are obtained as in the multiplicative case, but with the additive part as defined at the beginning of this section. Unfortunately, as mentioned in that remark (loc. cit.), the translation $(\ )^{\delta}$ which yields linear lambda terms (cf. Corollary 14) does not directly extend beyond the multiplicatives.

7 The Exponentials

What has been done for the additives can be extended to the exponentials: namely, we may extend the translations π defined for **MALL** with the following. (The reader is referred to the comments in Section 3.3.)

$$\pi\left(\overline{?A}\right) = z(\mathbf{wdc})\overline{\mathbf{w}}$$

$$\pi\left(\begin{array}{cc} \mathcal{R} & \\ A & \\ \cline{1-1} ?A & \Gamma \end{array}\right) = (\nu x)z(\mathbf{wdc})\bar{\mathbf{d}}\langle x\rangle\, P_x$$

$$\pi\left(\begin{array}{cc} \mathcal{R} & \\ ?A \quad ?A & \\ \cline{1-1} ?A & \Gamma \end{array}\right) = (\nu xy)(z(\mathbf{wdc})\overline{\mathbf{c}}\langle xy\rangle)\, P_{xy}$$

and

$$\pi\left(\begin{array}{c} \boxed{\begin{array}{c} \mathcal{R}_1 \\ ?\Gamma, \quad A \end{array}} \\ ?\Gamma \quad !A \end{array}\right) = !^x_z\, Q_{x\ \tilde{u}} =$$

$$= (\nu\mathbf{wdc})\overline{z}\langle\mathbf{wdc}\rangle \begin{cases} \mathbf{w}().(\|_{i\leq|\tilde{u}|}\ u_i(\mathbf{wdc})\bar{\mathbf{w}}) \\ \qquad + \\ \mathbf{d}(x).Q_{x\ \tilde{u}} \\ \qquad + \\ \mathbf{c}(z'z'').(\nu\tilde{u}'\tilde{u}'')(\|_{i\leq|\tilde{u}|}\ u_i(\mathbf{wdc})\overline{\mathbf{c}}\langle u_i', u_i''\rangle \parallel !^x_{z'}\, Q_{x\ \tilde{u}'} \parallel !^x_{z''}\, Q_{x\ \tilde{u}''}) \end{cases}$$

Exponential Symmetric Reductions:

The reduction from

$$\begin{array}{ccc} & \boxed{\begin{array}{c} \mathcal{R}_1 \\ A\ ?\Gamma \end{array}} & \\ \overline{?A^\perp} & !A & ?\Gamma \\ \cline{1-2} & & \vdots \end{array}$$

where $?\Gamma = ?C_1, \ldots, ?C_n$ to

$$\begin{array}{c} \overline{?C_1} \ldots \overline{?C_n} \\ \vdots \end{array}$$

corresponds to

$$(\nu z)\,(z(\mathbf{wdc})\overline{\mathbf{w}} \parallel \mathbf{!}^x_z Q_{x\ \tilde{u}}) \succ_* (\|_{i \leq |\tilde{u}|}\ u_i(\mathbf{wdc})\bar{\mathbf{w}})$$

using the recursive definition of $\mathbf{!}^x_z Q_{x\ \tilde{u}}$ and two ordinary reduction steps.

The reduction from

$$\begin{array}{ccc} \vdots & & \\ \dfrac{A^\perp}{?A^\perp} & & \boxed{\begin{array}{c} \mathcal{R}_1 \\ A\ ?\Gamma \end{array}} \\ \multicolumn{2}{c}{\underline{\hphantom{?A^\perp \qquad\qquad !A}}} & \\ & !A & ?\Gamma \\ & & \vdots \end{array}$$

to

$$\begin{array}{ccc} \vdots & \mathcal{R}_i & \\ \underline{A^\perp \qquad\qquad A} & & ?\Gamma \\ & & \vdots \end{array}$$

corresponds to

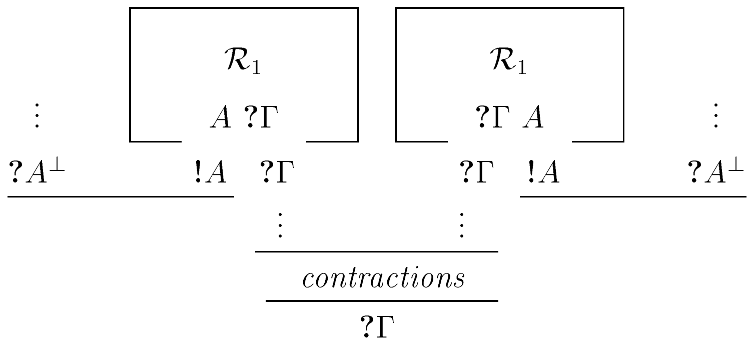
$$(\nu z)\left((\nu xy)(z(\mathbf{wdc})\bar{\mathbf{d}}\langle x\rangle)\,P_x \parallel \mathbf{!}^x_z Q_{x\ \tilde{u}}\right) \succ_* (\nu x)\,(P_x \parallel Q'_{x\ \tilde{u}})$$

using the recursive definition of $\mathbf{!}^x_z Q_{x\tilde{u}}$ and two ordinary reduction steps.

The reduction from

$$\begin{array}{ccc} \vdots & & \\ \dfrac{?A^\perp \quad ?A^\perp}{?A^\perp} & & \boxed{\begin{array}{c} \mathcal{R}_1 \\ A\ ?\Gamma \end{array}} \\ \multicolumn{2}{c}{\underline{\hphantom{?A^\perp \qquad\qquad !A}}} & \\ & !A & ?\Gamma \\ & & \vdots \end{array}$$

to



corresponds to

$$(\nu z)\,((\nu xy)(z(\mathbf{wdc})\bar{\mathbf{c}}\langle xy\rangle)\,P_{xy} \parallel \mathbf{!}_z^x Q_{x\ \tilde{u}}) \succ_* \\ (\nu \tilde{u}'\tilde{u}'')(\|_{i\leq|\tilde{u}|}\, u_i(\mathbf{wdc})\overline{\mathbf{c}}\langle u_i', u_i''\rangle(\nu xy)\Big(P_{xy} \parallel \mathbf{!}_x^{x'}\, Q_{x'\ \tilde{u}'} \parallel \mathbf{!}_y^{x''}\, Q_{x''\ \tilde{u}''})\Big)$$

using the recursive definition of $\mathbf{!}_z^x Q_{x\ \tilde{u}}$ and two ordinary reduction steps.

Thus the above reductions are sound with respect to the reduction strategy that applies symmetric reductions outside all boxes. Moreover, *if rewriting using the recursive definition of $\mathbf{!}_z^x Q_{x\ \tilde{u}}$ can be done without producing deadlocked terms–hence, without garbage collection*, then we can obtain local fullness as before, since the summands in the right-hand side of that definition are guarded by their prefixes, as in the additive case. As mentioned earlier, this would seem to depend on how the rewriting theory of the recursive definition of π-terms such as $\mathbf{!}_z^x Q_{x\ \tilde{u}}$ is implemented. The latter awaits further development of the synchronous π-calculus. Moreover, for now, nothing can be said about local fullness of the above translation until details of such an implementation are given.

8 Acknowledgements

We have benefitted from numerous discussions with colleagues during the preparation of this paper. We would particularly like to thank Samson Abramsky and Robin Milner for their continuing support and interest, as well as many thoughtful comments. Also special thanks to Mads Dam, Davide Sangiorgi, and David Turner for helpful discussions on many aspects of concurrency theory, and to an anonymous referee for his insightful comments.

G. Bellin wants to thank Gordon Plotkin and Robin Milner for their kind invitation and support at the Laboratory for Computer Science, Edinburgh during 1990-92 . P. J. Scott would also like to thank LFCS, its director G. Plotkin, and the Department of Computer Science, University of Edinburgh, for their kind hospitality during his Sabbatical (1991-92), when this work was carried out.

References

[1] S. Abramsky. Computational Interpretations of Linear Logic, *Theoretical Computer Science* , 1993.

[2] S. Abramsky. Proofs-as-processes. Seminar talk, Dept. of Computer Science, University of Edinburgh, 1992.

[3] S. Abramsky and R. Jagadeesan. New Foundations for the Geometry of Interaction, IEEE Symposium on Logic in Computer Science (LICS), 1992, pp. 211-222.

[4] S. Abramsky and R. Jagadeesan. Games and Full Completeness Theorem for Multiplicative Linear Logic. To appear.

[5] A. Asperti. Causal Dependencies in Multiplicative Linear Logic with MIX, manuscript.

[6] G. Bellin. Proof Nets for Multiplicative and Additive Linear Logic, Report LFCS-91-161, May 1991, Dept. of Computer Science, Univ. of Edinburgh.

[7] G. Bellin. Mechanizing Proof Theory: Resource-Aware Logics and Proof-Transformations to Extract Implicit Information, Phd Thesis, Stanford University. Available as: Report CST-80-91, June 1990, Dept. of Computer Science, Univ. of Edinburgh.

[8] G. Bellin and J. van de Weile. Proof Nets and Typed Lambda Calculus, in preparation.

[9] A. Blass. A game semantics for linear logic. *Annals of Pure and Applied Logic*, **56**, 1992, pp. 183-220.

[10] R. Blute. Linear logic, coherence and dinaturality, *Th. Computer Science* **115** (1993), pp. 3-41.

[11] V. Danos. La logique linéaire appliquée à l'étude de divers processus de normalisation et principalement du λ-calcul, Thèse de doctorat, U. Paris VII, April 1990.

[12] V. Danos and L. Regnier. The Structure of Multiplicatives, *Arch. Math. Logic* (1989) **28**, pp. 181-203.

[13] J-Y. Girard. Linear Logic, *Theoretical Computer Science* **50**, 1987, pp. 1-102.

[14] J-Y. Girard. Multiplicatives, *Rend. Sem. Matem. (Torino)* , 1987 (Special Issue: Logic and Computer Science, New Trends and Applications, Oct. 1986), pp. 11-33.

[15] J-Y. Girard. Linear Logic and Parallelism, manuscript, 1987.

[16] J-Y. Girard. Towards a Geometry of Interaction, in: *Categories in Computer Science and Logic*, ed. by J.W. Gray and A. Scedrov, *Contemp. Math*, **92**, AMS, 1989, pp. 69-108.

[17] J-Y.Girard. Geometry of Interaction I: Interpretation of system F, in: *Logic Colloquium '88* , ed. R. Ferro, et al. North-Holland, 1989, pp. 221-260.

[18] J-Y.Girard. Geometry of Interaction 2: Deadlock-free Algorithms. COLOG-88 (P. Martin-Löf, G. Mints, eds.) Springer LNCS, **417**, 1990, pp. 76-93.

[19] J-Y. Girard, Y.Lafont, and P. Taylor. *Proofs and Types*, Cambridge Tracts in Theoretical Computer Science **7** , Cambridge University Press, 1989.

[20] J-Y. Girard, A. Scedrov, and P.J. Scott, Bounded Linear Logic, *Theoretical Computer Science*,**97**, No. 1, pp.1-66.

[21] G. Gonthier, M. Abadi, and J.J.Levy. Linear logic without boxes. In: *Proc. of the Seventh Symposium on Logic in Computer Science (LICS)*, Comp Soc. Press, IEEE, 1992.

[22] P. Malacaria and L. Regnier. Some Results on the Interpretation of λ-calculus in Operator Algebras, Logics in Computer Science (LICS), 1991, Amsterdam, IEEE Computer Press, pp. 63-72.

[23] D. Miller. The π-calculus as a theory in linear logic: Preliminary results, Proc. 1992 Workshop on Extensions to Logic Programming, Springer LNCS **660**, eds. E. Lamma and P. Mello, 1993, pp. 242-265.

[24] R. Milner. Personal notes.

[25] R. Milner. The polyadic π-Calculus: a tutorial, in: *Logic and Algebra of Specification, eds. F.L Bauer, W. Brauer, and H. Schwichtenberg* , Springer-Verlag, 1993.

[26] R. Milner. Turing Award Lecture, 1992.

[27] R. Milner. The Synchronous π-Calculus. Unpublished lectures, Dept. of Computer Science, University of Edinburgh, 1992.

[28] R. Milner. Action Structures and the π-calculus, to appear in: Proceedings of NATO Advanced Study Institute, *Proof and Computation (held at Marktoberdorf)*, 1993. (Also available as an LFCS report, Dept. of Computer Science, University of Edinburgh.)

[29] E. Monteiro. Linear Logic as CSP, to appear in: *J. Logic and Computation.*

[30] L. Regnier. Lambda-calcul et réseaux. Thèse de doctorat, U. Paris VII, Mathématiques, 1992.

[31] C. Retoré. *Résequx et Séquents Ordonnés* Thèse de doctorat, U. Paris VII, Mathématiques, 1993.

[32] A. S. Troelstra. Lectures on Linear Logic, CSLI Lecture Notes (V. 29), Center for the Study of Language and Information, Stanford University, 1992.

Gianluigi Bellin
Equipe de Logique mathématique
Université Paris VII
2 Place Jussieu
75251 Paris cedex 05
France
bellin@logique.jussieu.fr

Philip J. Scott
Dept. of Mathematics
Univ. of Ottawa
585 King Edward
Ottawa, Ont.
Canada K1N 6N5
scpsg@acadvm1.uottawa.ca