

ACYCLIC SOLOS AND DIFFERENTIAL INTERACTION NETS

THOMAS EHRHARD AND OLIVIER LAURENT

Preuves Programmes Systèmes, CNRS – Université Paris 7, France
e-mail address: Thomas.Ehrhard@pps.jussieu.fr, Olivier.Laurent@ens-lyon.fr

ABSTRACT. We present a restriction of the solos calculus which is stable under reduction and expressive enough to contain an encoding of the pi-calculus. As a consequence, it is shown that equalizing names that are already equal is not required by the encoding of the pi-calculus. In particular, the induced solo diagrams bear an acyclicity property that induces a faithful encoding into differential interaction nets. This gives a (new) proof that differential interaction nets are expressive enough to contain an encoding of the pi-calculus.

All this is worked out in the case of finitary (replication free) systems without sum, match nor mismatch.

INTRODUCTION

The question of extending the Curry-Howard correspondence (between the λ -calculus and intuitionistic logic) to concurrency theory is a long-standing open problem. We developed in a previous paper [EL10] a translation between the π -calculus [MPW92] and differential interaction nets [ER06]. This has shown that differential linear logic — a logical system whose sequent calculus [EL10] has been obtained by the first author from a precise analysis of some denotational models of linear logic based on vector spaces [Ehr02] — is a reasonable candidate for a Curry-Howard correspondence with concurrent computation, since differential interaction nets appear to be expressive enough to represent key concurrency primitives as provided by the π -calculus.

There are other works trying to relate variants of linear logic proof-nets with process algebras (such as [Abr93, BS94, BM06, FM05, Maz05, HL10]) or to work on graphical syntaxes for encoding the π -calculus (see [Mil94, LPV01, JM04] for example). However, even if some of them try to build bridges with (linear) logic, they require some ad hoc graphical constructions without logical foundations. Our goal is to build on differential interaction nets coming with the logical justification of differential linear logic.

1998 ACM Subject Classification: F.3.2, F.4.1, F.1.2.

Key words and phrases: solos calculus, pi-calculus, prefix, typing, differential interaction nets.

This work has been partially funded by the French ANR projet blanc “Curry Howard pour la Concurrency” CHOCO ANR-07-BLAN-0324.

Let us tell a bit more about the genesis of our translation. We discovered the notion of *communication areas* as particular differential interaction nets able to represent some communication primitives. However, due to the asynchronous flavour of differential interaction nets, it was not immediate that additional constructions such as prefixing could be easily encoded. This led us to have a look at the solos calculus.

The *solos calculus* [LV03] has allowed to prove how action prefixes (thus sequentiality constraints) can be encoded in a calculus without prefixes (results of this kind also appear in [HY94] and are going much further than in the asynchronous π -calculus [HT91, Bou92]). This is done by encoding the π -calculus into the solos calculus [LV03]. From this point of view, the solos calculus can be seen as a low level concurrent language: even more basic than the π -calculus.

The main goal of the present paper is to stress the very close relation which appears between *differential interaction nets* [ER06] (the graphical syntax for differential linear logic) and *solo diagrams* [LPV01] (the graphical syntax for the solos calculus).

Following the ingredients of [EL10], it is possible to give a simple representation of solo diagrams as differential interaction nets (based on an implementation of nodes as communication areas). Such a “static” correspondence is not very interesting if there is no “dynamic” counterpart in the reduction semantics. And this is where troubles come into the picture... The reduction semantics *almost* match... There is a mismatch between the solos calculus and communication areas with respect to the identification of an occurrence of name with another occurrence of the same name during reduction (let us call this phenomenon, *self-identification*). Note that such an identification never occurs in the π -calculus. Since name passing is handled by the substitution of a bound name by *another* name. As in the *fusion calculus* [PV98], the solos calculus defines communication by unification of names, so that two occurrences of a name could be identified by reduction. If x has to be identified with itself, it is just considered as a dummy operation in a unification setting since we already have $x = x$. The situation is different with differential interaction nets since communication areas keep track of such an identification through an explicit link connecting the communication area associated with x to itself. Reduction is not able to simplify such a link in order to give back a communication area. One gets some more complicated structure.

In [EL10] we were directly working with the π -calculus (thus with name passing by substitution, thus without self-identification). This way a direct interpretation of the whole (finitary) π -calculus into differential interaction nets was possible. The approach we propose here is different: we look for a restriction of the solos calculus (avoiding self-identification) for which the static correspondence with differential interaction nets also works at the dynamic level. This is the *acyclic solos calculus*. This requires us to restrict the source calculus but allows us to deal with a very natural translation.

The acyclic solos calculus is obtained from a precise analysis of the translation of the π -calculus into the solos calculus. The main idea is to be able to isolate, inside the unification mechanism which is a symmetric operation (when one unifies a name x with a name y), a “flow” from one side to the other (from x to y for example) as it happens in a substitution mechanism (in the reduction of $\bar{u}(x).P \mid u(y).Q$ in the π -calculus, one can consider a flow going from x to y , with the binding occurrence y “controlling” all the other occurrences of y in Q). The first ingredient in our definition of the acyclic solos calculus is a simple typing system allowing us to label all the object occurrences of names of a solos term with

two *protocols* (S (send) and R (receive)) in a kind of uniform way. The main consequence of the typing system is to break symmetry in reduction/unification: when two occurrences of names have to be unified one is an S-occurrence and the other one is an R-occurrence. However this does not impose important structural or behavioural constraints on protocols. This is given by the second part of the definition of the calculus: the *acyclicity conditions* (AC1–AC5). The justification for these conditions is to mimic the structure of the input prefix $u(y).Q$ of the π -calculus inside the solos calculus: exactly one binding occurrence y (this will be the R-occurrence of the name y in the solos term), all the other occurrences (thus S-occurrences for object occurrences in acyclic solos) are coming “sequentially after” the binding one, ... Informally, all this together leads to a name passing by unification which is very close to a substitution mechanism if we understand the unification of an S-occurrence with an R-occurrence as a substitution flow from the S-occurrence to the R-occurrence (and then all the other occurrences of this name in S-position). Formally, we prove that self-identification never occurs in the acyclic solos calculus.

We also show that the translation of the (finitary) π -calculus into the solos calculus following [LV03] has its range contained in the acyclic solos calculus. This proves that our restriction of the solos calculus still has a reasonable expressive power. Moreover this gives us an alternative proof (with respect to the work presented in [EL10]) of the existence of a translation of the finitary π -calculus into differential interaction nets.

The first part of the paper is devoted to the introduction of the required calculi (π -calculus and solos calculus in Section 1, solo diagrams in Section 2, and differential interaction nets in Section 3 which is almost copy-pasted from [EL10]). In Section 4 we elaborate on the material presented in [EL10] to define the simple static translation of solo diagrams [LPV01] (the graphical syntax for the solos calculus) into differential interaction nets. We discuss the problems arising at the dynamic level, and we give a sufficient condition on solo diagrams for the translation into differential interaction nets to be a bisimulation: an occurrence of a name should never be unified with another occurrence of the same name (*i.e.* no self-identification).

The main technical contribution of the paper comes in the last part (Section 5). Since the property that there is no self-identification in the redexes of a solos term is of course *not* preserved under the reduction of solos, we have to find a more clever property. We define the *acyclic solos calculus* through a typing system for assigning protocols and the acyclicity conditions (AC1–AC5). We prove this restriction to be well behaved with respect to the reduction of the solos calculus. We show that the translation of a π -term is always an acyclic solos term, showing the expressiveness of the system. Finally we prove that the sufficient condition introduced in Section 4 is fulfilled by solo diagrams corresponding to acyclic solos terms, showing that we obtain a bisimulation with respect to differential interaction nets.

1. THE π -CALCULUS AND THE SOLOS CALCULUS

In this section we recall the definition of the π -calculus and of the solos calculus we are going to use. We also recall the translation from π to solos given in [LV03].

In the π -calculus [MPW92], the two basic components of communication are the output and input prefixes $\bar{u}(x).P$ and $v(y).Q$ (with occurrences of y bound in Q). Communication can occur if $u = v$ and in this case, it is obtained by substituting y by x (in Q). Moreover these prefix constructions induce a sequential behaviour: the continuation P (resp. Q)

cannot interact with other agents before the output on u (resp. the input on v) has been triggered.

The fusion calculus [PV98] provides a generalization with communication modeled by unification. This makes sending and receiving perfectly symmetric and thus the distinction between $\bar{u}x.P$ and $ux.P$ is purely formal. Interaction is just constrained to occur between two dual entities u and \bar{u} . The symmetry is broken when one translates π -terms into fusion terms by: $u(x).P \mapsto (x)ux.P$ (and unification restrained to the image of this translation becomes substitution by properties of the binding restriction $(x)P$). Notice that the sequentiality aspect of prefixing remains.

Based on the unification mechanism provided by the fusion calculus, the solos calculus [LV03] is free from explicit sequentiality constructions: the fusion prefixes are restricted to dummy continuations $\bar{u}x.0$ and $ux.0$ (where 0 is an inactive process). It is shown in [LV03] that dyadic solos $\bar{u}xy$ and uxy are as expressive as the whole fusion calculus.

Comments on chosen calculi. Since our goal is to focus on prefixing and sequentiality, we deal with calculi without replications nor recursive definitions, without match/mismatch and without sums (see Conclusion for additional comments).

We do not want to spend time to deal with arbitrary arities in the calculi we consider. This is why we only consider *monadic* π -terms. There are three reasons for that: it makes the presentation simpler, it does not lead to a loss of expressiveness, and finally the polyadic case has already been considered in [EL10]. As a consequence (see the translation in Section 1.3), we are led to consider a *triadic* solos calculus (all the names are of arity exactly 3) and *triadic* solo diagrams (all the multiedges are of arity exactly 3). The more general case of arbitrary arities could easily be obtained by introducing the appropriate sortings on the various calculi.

1.1. The π -calculus. The terms of the (monadic, finitary) π -calculus are given by:

$$P ::= 0 \mid u(x).P \mid \bar{u}\langle x \rangle.P \mid (P \mid P) \mid \nu x.P$$

where both $u(x).P$ and $\nu x.P$ bind occurrences of x in P .

The *structural congruence* on π -terms is the least congruence containing α -equivalence and:

$$\begin{aligned} 0 \mid P &\equiv P \\ P \mid Q &\equiv Q \mid P \\ (P \mid Q) \mid R &\equiv P \mid (Q \mid R) \\ \nu x.\nu y.P &\equiv \nu y.\nu x.P \\ \nu x.0 &\equiv 0 \\ (\nu x.P) \mid Q &\equiv \nu x.(P \mid Q) && \text{if } x \notin \text{fn}(Q) \end{aligned}$$

The *reduction semantics* of the π -calculus is given by:

$$\frac{}{\bar{u}\langle x \rangle.P \mid u(y).Q \rightarrow P \mid Q[x/y]} \quad \frac{P \rightarrow Q}{P \mid R \rightarrow Q \mid R} \quad \frac{P \rightarrow Q}{\nu x.P \rightarrow \nu x.Q} \quad \frac{P \equiv P' \quad P' \rightarrow Q' \quad Q' \equiv Q}{P \rightarrow Q}$$

1.2. **The solos calculus.** Introduced in [LV03], the goal of the solos calculus is to prove the expressiveness of a calculus without prefix construction.

The terms of the (triadic) solos calculus are given by:

$$P ::= 0 \mid u x_1 x_2 x_3 \mid \bar{u} x_1 x_2 x_3 \mid (P \mid P) \mid (x) P$$

where $(x) P$ binds occurrences of x in P .

The *structural congruence* is the least congruence containing α -equivalence and:

$$\begin{aligned} 0 \mid P &\equiv P \\ P \mid Q &\equiv Q \mid P \\ (P \mid Q) \mid R &\equiv P \mid (Q \mid R) \\ (x)(y) P &\equiv (y)(x) P \\ (x) 0 &\equiv 0 \\ ((x) P) \mid Q &\equiv (x)(P \mid Q) && \text{if } x \notin \text{fn}(Q) \end{aligned}$$

This equivalence allows us to present terms in the solos calculus in canonical forms: either the 0 process or a bunch of scope constructions $(x_1)(x_2)\dots$ followed by solos in parallel.

The *reduction semantics* of the solos calculus is given by (\tilde{z} stands for $z_1 \dots z_n$):

$$\frac{}{(\tilde{z})(\bar{u} x_1 x_2 x_3 \mid u y_1 y_2 y_3 \mid P) \rightarrow P \sigma}$$

where σ is a most general unifier of $x_1 x_2 x_3$ and $y_1 y_2 y_3$, such that exactly the names w in \tilde{z} are modified, *i.e.* satisfy $\sigma(w) \neq w$ (in particular, in each equivalence class of names induced by unification, at most one name is free).

$$\frac{P \rightarrow Q}{P \mid R \rightarrow Q \mid R} \quad \frac{P \rightarrow Q}{(x) P \rightarrow (x) Q} \quad \frac{P \equiv P' \quad P' \rightarrow Q' \quad Q' \equiv Q}{P \rightarrow Q}$$

An alternative (but equivalent) definition of this reduction semantics is given in [LV03] together with additional explanations.

For example, in $(x)(y)(z)(w)(\bar{u} uxy \mid u zww \mid \bar{v} zu y)$, the only possible reduction is between $\bar{u} uxy$ and $u zww$. It induces the identifications $u = z$, $x = w$ and $y = w$, thus two equivalence classes $\{u, z\}$ and $\{x, y, w\}$. In the first one, u is free and thus the only possibility is to map z to u . In the second one, all the elements are bound, we choose one of them: y for example (the other choices would lead to structurally congruent results). We consider the unifier containing $z \mapsto u$, $x \mapsto y$, $w \mapsto y$ and which is the identity on the other names. We obtain the reduct $(y)\bar{v} uuy$.

1.3. **From π -terms to solos.** In [LV03], the authors give different translations of the fusion calculus [PV98] into solos. We are going to focus on one of them (the one which does not introduce matching). By pre-composing this translation with the canonical embedding of the π -calculus into the fusion calculus: $u(x).P \mapsto (x) u x.P$, we obtain the translation of the π -calculus into solos that we present here.

A π -term P is translated as $[P]$:

$$\begin{aligned}
C_v &:= (z) v z z v \\
[0]_v &:= 0 \\
[u(x).P]_v &:= (w) (y) (\bar{v} u w y \mid C_y \mid (x) (v') (w x v v' \mid [P]_{v'})) \\
[\bar{u}(x).P]_v &:= (w) (y) (\bar{v} u w y \mid C_y \mid (v') (\bar{w} x v' v \mid [P]_{v'})) \\
[P \mid Q]_v &:= [P]_v \mid [Q]_v \\
[\nu x.P]_v &:= (x) [P]_v \\
[P] &:= (v) ([P]_v \mid C_v)
\end{aligned}$$

As shown in [LV03], this encoding is adequate with respect to weak barbed congruence \approx : $[P] \approx [Q]$ implies $P \approx Q$.

The π -term $\nu x.(\bar{u}(x).0 \mid x(y).0) \mid u(z).\bar{z}(t).0$ is translated as a solos term which is structurally congruent to $(v) ((x) (P \mid Q) \mid R \mid C_v)$ with:

$$\begin{aligned}
P &= (w) (y) (\bar{v} u w y \mid C_y \mid (v') \bar{w} x v' v) \\
Q &= (w) (y') (\bar{v} x w y' \mid C_{y'} \mid (y) (v') w y v v') \\
R &= (w) (y) (\bar{v} u w y \mid C_y \mid (z) (v') (w z v v' \mid (w) (y) (\bar{v}' z w y \mid C_y \mid (v'') \bar{w} t v'' v)))
\end{aligned}$$

By applying reduction, we obtain in particular the following reducts (up to structural congruence):

$$\begin{aligned}
&(v) ((x) ((C_v \mid (v') \bar{u} x v' v) \mid Q) \mid R) \\
&(v) ((x) (((v') \bar{u} x v' v) \mid Q) \mid C_v \mid (z) (v') (u z v v' \mid (w) (y) (\bar{v}' z w y \mid C_y \mid (v'') \bar{w} t v'' v))) \\
&\quad (v) (x) (Q \mid C_v \mid ((w) (y) (\bar{v} x w y \mid C_y \mid (v'') \bar{w} t v'' v))) \\
&\quad (v) (x) (C_v \mid (y) (v') x y v v' \mid ((w) (y) (\bar{v} x w y \mid C_y \mid (v'') \bar{w} t v'' v))) \\
&\quad (v) (x) ((y) (v') x y v v' \mid C_v \mid (v'') \bar{x} t v'' v) \\
&\quad (v) C_v
\end{aligned}$$

2. SOLO DIAGRAMS

To make the relation with differential interaction nets simpler, we go through the graphical syntax associated with the solos calculus: solo diagrams [LPV01].

A (triadic) *solo diagram* is given by a finite set of nodes and a finite multiset of (ternary) multiedges (directed edges with a list of three nodes as source and one node as target). A node is tagged as either *free* or *bound*. A multiedge is tagged as either *input* or *output*. Any node must be a source or a target of a multiedge.

2.1. Reduction. The reduction of solo diagrams is given:

- by choosing two multiedges e_1 and e_2 of opposite polarities (one input and one output) with the same target (we call them *dual multiedges*) and with respective sources $[\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3]$ and $[\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$,

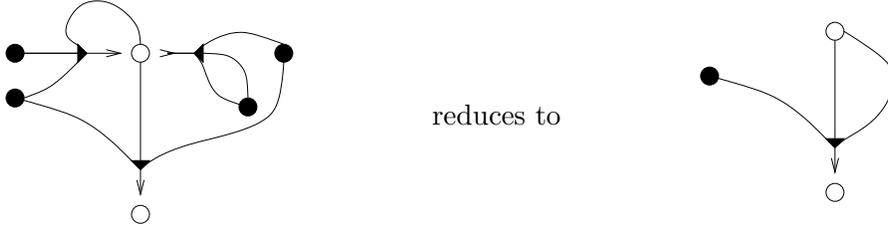


Figure 1: A reduction in solo diagrams.

- and by identifying the two nodes \mathbf{n}_1 and \mathbf{m}_1 , the two nodes \mathbf{n}_2 and \mathbf{m}_2 , and the two nodes \mathbf{n}_3 and \mathbf{m}_3 . The reduction is allowed to occur only if it does not identify two free nodes. Moreover when a free node is identified with a bound node, the obtained node is free and when two bound nodes are identified, the obtained node is bound.

The chosen multiedges are erased and nodes which are not anymore source or target of a multiedge are also removed. This can be applied to both free and bound nodes.

During a reduction step, the number of multiedges decreases by two and the number of nodes decreases or remains the same.

In the graphical representation, we draw free nodes as white dots, bound nodes as black dots, output edges with an outgoing arrow and input edges with an ingoing arrow. An example is given in Figure 1.

2.2. Relation with solos. A term of the solos calculus can easily be translated into a solo diagram. Free names are translated as free nodes, bound names as bound nodes and solos as multiedges:

- The 0 term is translated as the empty graph with no edge.
- The solo $u x_1 x_2 x_3$ is translated as the graph with free nodes corresponding to elements of $\{u, x_1, x_2, x_3\}$, and with one input multiedge with source $[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3]$ and target \mathbf{u} (where \mathbf{x}_i is the node corresponding to the name x_i and \mathbf{u} is the node corresponding to the name u).
- The solo $\bar{u} x_1 x_2 x_3$ is translated in the same way with an output multiedge.
- The parallel composition $P \mid Q$ is obtained by “graph union”: union of the sets of nodes (*i.e.* nodes corresponding to the same name are identified) and sum of the multisets of multiedges.
- The restriction $(x)P$ corresponds to turning the free node \mathbf{x} corresponding to the name x (if any) into a bound node.

As shown in [LPV01], two solos terms are structurally congruent if and only if the corresponding solo diagrams are isomorphic. Moreover reduction in solo diagrams reflects faithfully the reduction of the solos calculus.

The two solo diagrams of Figure 1 correspond to $(x)(y)(z)(w)(\bar{u} uxy \mid u zww \mid \bar{v} zu y)$, respectively, $(y)\bar{v} uuy$ (the examples used in Section 1.2).

2.3. Labeled transition system. We follow the same methodology as in [EL10] by using labels to distinguish the actions of a process. In a term syntax as in [EL10], labels are put on prefixes (or solos). In the graphical syntax of solo diagrams, we put labels on multiedges (since multiedges correspond to solos).

We fix a countable set \mathcal{L} of labels to be used for all our labeled transition systems.

A solo diagram is *labeled* if its multiedges are equipped with different labels belonging to \mathcal{L} .

Definition 2.1 (The transition system $\mathbb{S}_{\mathcal{L}}$). The objects of the *labeled transition system* $\mathbb{S}_{\mathcal{L}}$ are labeled solo diagrams and transitions are labeled by pairs of distinct elements of \mathcal{L} . Let G and H be two labeled solo diagrams, $G \xrightarrow{lm} H$ if H is obtained from G by applying a reduction step to the input multiedge labeled l and to the output multiedge labeled m (the labels of the remaining multiedges of H must be the same as for the corresponding ones in G).

This is quite different from what is usually done for defining transition systems for a process algebra [MPW92]. The standard approach aims at analyzing the possible interactions of a process with its environment. Then bisimulation is used to show that two processes have the same “external” behaviour. Our goal is to use bisimulation to compare the internal behaviours of solo diagrams and of differential interaction nets. We want to illustrate that differential interaction nets are sufficiently expressive for simulating concurrency and mobility. This requires the transition systems we use for bisimulation to carry precise informations about reductions of processes, in particular about the sub-entities taking part into the interactions.

2.4. Solo diagrams with identifications. In order to compare solo diagrams with differential interaction nets, it will be useful to decompose the reduction of solo diagrams by introducing the notion of *solo diagrams with identifications*.

Definition 2.2 (Solo diagram with identifications). *Solo diagrams with identifications* are solo diagrams equipped with a finite set of undirected edges (usual binary edges, not multiedges, which connects nodes of the solo diagram). These edges are called *identification edges*.

We can decompose the reduction of solo diagrams we have presented above by using identification edges. A reduction step for a solo diagram G is defined as follows:

- (R1) choose two dual multiedges e_1 and e_2 (*i.e.* of opposite polarities and with the same target) with respective sources $[\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3]$ and $[\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$;
- (R2) build the solo diagram with identifications $G[e_1, e_2]$ obtained from G by erasing e_1 and e_2 and by introducing three identification edges: between \mathbf{n}_1 and \mathbf{m}_1 , between \mathbf{n}_2 and \mathbf{m}_2 , and between \mathbf{n}_3 and \mathbf{m}_3 ;
- (R3) contract the graph $G[e_1, e_2]$ by (repeatedly) choosing an identification edge and by identifying the two (or one) nodes it connects if at least one of them is bound.

The reduction succeeds (*i.e.* is a valid reduction of solo diagrams) if we reach a solo diagram with no remaining identification edge. It means in particular that step (2) does not introduce identification edges between two free names.

In the graphical representation, we draw identification edges as dashed edges.

If we refine the reduction of the example of Figure 1 by means of solo diagrams with identifications, the results of step (2) and then of one application of step (3) are in Figure 2.



Figure 2: An (3) reduction step in solo diagrams with identifications.

3. DIFFERENTIAL INTERACTION NETS

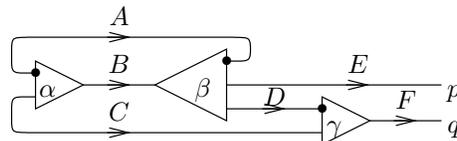
3.1. The general formalism of interaction nets. We recall the general (graphical) syntax of interaction nets, as introduced in [Laf95]. See also [Maz06, ER06] for more details.

Statics of interaction nets. We first define a general typed syntax of interaction nets.

Assume we are given a set of *symbols* and that an arity (a non-negative integer) and a typing rule is associated with each symbol. This typing rule is a list (A_0, A_1, \dots, A_n) of types, where n is the arity associated with the symbol. Types are formulae of some system of linear logic (in particular if A is a type, A^\perp is also a type and $A^{\perp\perp} = A$). An *interaction net* is made of *cells*. With each cell γ is associated exactly one symbol and therefore an arity n and a typing rule (A_0, A_1, \dots, A_n) . Such a cell γ has one *principal port* p_0 and n *auxiliary ports* p_1, \dots, p_n . An interaction net has also a finite set of *free ports*. All these ports (the free ports and the ports associated with cells) have to be pairwise distinct and a set of *wires* is given. This wiring is a set of pairwise disjoint sets of ports of cardinality 2 (ordinary wires) or 0 (loops¹), and the union of these wires must be equal to the set of all ports of the interaction net. In other words, each port of the interaction net (free or associated with a cell) is connected to exactly one other port (free or associated with a cell) through a wire, and each such wire connects exactly two ports: ports cannot be shared. The free ports of the interaction net are those which are not associated with a cell.

An *oriented wire* of the interaction net is an ordered pair (p_1, p_2) where $\{p_1, p_2\}$ is a wire. In an interaction net, a type is associated with each oriented wire, in such a way that if A is associated with (p_1, p_2) , then A^\perp is associated with (p_2, p_1) . Last, the typing rules of the cells must be respected in the sense that for each cell γ of arity n , whose ports are p_0, p_1, \dots, p_n and typing rule is (A_0, A_1, \dots, A_n) , denoting by p'_0, p'_1, \dots, p'_n the ports of the interaction net uniquely defined by the fact that the sets $\{p_i, p'_i\}$ are wires (for $i = 0, 1, \dots, n$), then the oriented wires $(p_0, p'_0), (p'_1, p_1), \dots, (p'_n, p_n)$ have types A_0, A_1, \dots, A_n respectively. The free ports of the interaction net constitute its *interface*. With each free port p can be associated the type of the unique oriented wire whose endpoint is p : this is the type of p in the interface of the interaction net.

Here is a typical example of a typed interaction net:



¹To be more precise, one has to specify the number of loops in the net, but this will not play any role in the sequel.

with cells of symbols α , β and γ , of respective types (B, A^\perp, C^\perp) , $(B^\perp, A, E^\perp, D^\perp)$ and (F, D, C) . The interface is $(p : E, q : F)$. Cells are represented as triangles, with principal port located at one of the angles and other ports on the opposite edge. We often draw a black dot to locate the auxiliary port number 1.

Dynamics of interaction nets. In Lafont’s traditional interaction nets, a reduction rule associates, to each pair of cells connected through their principal ports, an interaction net with the same interface as this two-cells net. Moreover at most one rule is given for each pair of cells. The application of a reduction rule substitutes (inside an interaction net) a pair of cells connected through their principal ports by the associated reduct.

These ideas are strongly related with key steps of cut elimination in logics, where two rules interact through their principal formulas.

Extensions of interaction nets. Since interaction nets have strong confluence properties, extensions are required to take concurrency and non-determinism into account [Ale99].

A possible way of extending interaction nets for more concurrent behaviours is to modify statics by introducing cells with many principal ports [Ale99, Kha03, Maz05] (this breaks simple logical interpretations by contradicting the usual property that a logical rule of the sequent calculus introduces *one* principal formula). In differential interaction nets, we stick to the traditional statics syntax (but act on the dynamics side).

A possible extension of the dynamics towards non-deterministic behaviours is to allow a choice between different possible reduction rules for a given pair of cells [Ale99].

We make a choice in-between this proposal and Lafont’s one (*i.e.* between many possible rules, and at most one rule). We consider only one reduction rule for each pair of cells but we extend interaction nets to formal sums of interaction nets [ER06]. This allows us to represent, as one reduction, a finite set of non-deterministic choices: when the reduct of the interaction of two cells is a (proper) sum of interaction nets.

3.2. Presentation of the cells. We define *differential interaction nets* from the interaction nets defined above. We first present our specific cells.

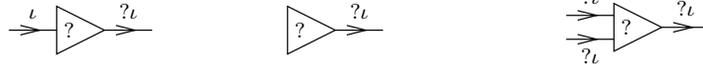
The untyped lambda-calculus can be seen as a typed lambda-calculus equipped with a recursive equation $D = D \Rightarrow D$. Following this idea, V. Danos and L. Regnier [Reg92] defined a notion of untyped proof-nets based on a single type symbol o (the type of outputs), subject to the following recursive equation $o = ?o^\perp \wp o$. Our nets will be typed using this typing system. We set $\iota = o^\perp$, so that $\iota = !o \otimes \iota$ and $o = ?\iota \wp o$. The tensor connective is used only with premises $!o$ and ι and dually for the par, and therefore, the only types we actually need are o , ι , $!o$ and $? \iota$ for typing our nets.

In the present setting, there are ten symbols: par (arity 2), bottom (arity 0), tensor (arity 2), one (arity 0), dereliction (arity 1), weakening (arity 0), contraction (arity 2), codereliction (arity 1), coweakening (arity 0) and cocontraction (arity 2). We present now the various cell symbols, with their typing rules, in a pictorial way.

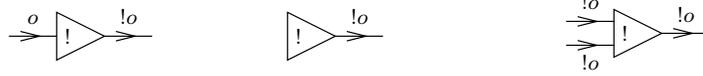
Multiplicative cells. The *par* and *tensor* cells, as well as their “nullary” versions *bottom* and *one* are as follows:



Exponential cells. They are typed according to a strictly polarized discipline. Here are first the *why not* cells, which are called *dereliction*, *weakening* and *contraction*:



and then the *bang* cells, called *codereliction*, *coweakening* and *cocontraction*:



Differential interaction nets. We define differential interaction nets from the cells above.

- A *simple differential interaction net* is a typed interaction net, which uses the multiplicative and exponential cells introduced above.
- A *differential interaction net* is a finite formal sum $S = s_1 + \dots + s_n$ ($n \geq 0$) of simple differential interaction nets having all the same interface, and this interface is then considered as the interface of S . A particular case is the differential interaction net $S = 0$ (the empty sum), and this differential interaction net has to be given together with its interface: there is a 0 net for each interface.

Labeled differential interaction nets. We now introduce labeled differential interaction nets, which are differential interaction nets where particular cells are equipped with labels. The labeled transition system of differential interaction nets will be defined using these labels in Section 3.3.3.

Let \mathcal{L}_τ be the countable set of labels obtained by adding a distinguished element τ (to be understood as the absence of label) to \mathcal{L} (introduced in Section 2.3). A *labeled simple differential interaction net* is a simple differential interaction net where all dereliction and codereliction cells are equipped with labels belonging to \mathcal{L}_τ . Moreover, if l and l' are two labels appearing in a labeled simple differential interaction net, either $l \neq l'$ or $l = l' = \tau$.

In our pictures, the labels of dereliction and codereliction cells will be indicated, when this label is different from τ . When its label is τ , a (co)dereliction cell will be drawn without any label.

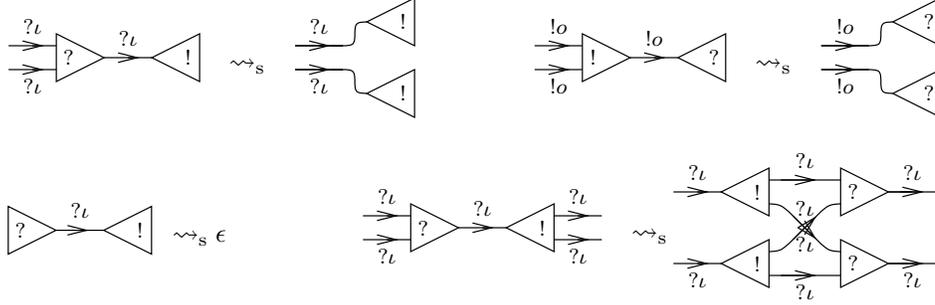
All the differential interaction nets we consider in this paper are labeled. In the sequel, since no confusion with other kinds of interaction nets will be possible, we shall use “net” for “labeled differential interaction net” and “simple net” for “labeled simple differential interaction net”.

3.3. Reduction rules. We denote by Δ the collection of all simple nets, ranged over by the letters s, t, u , with or without subscripts or superscripts, and by $\mathbb{N}\langle\Delta\rangle$ the collection of all nets (finite sums of simple nets with the same interface, including the empty sum), ranged over by the letters S, T, U , with or without subscripts or superscripts. We consider Δ as a subset of $\mathbb{N}\langle\Delta\rangle$ ($s \in \Delta$ is identified with the sum made of exactly one copy of s).

A *reduction rule* is a subset \mathcal{R} of $\Delta \times \mathbb{N}\langle\Delta\rangle$ consisting of pairs (s, S) where s is a simple net made of two cells connected by their principal ports and S is a net that has the same interface as s . There are actually reduction rules which transform simple nets into non-simple ones.

This set \mathcal{R} can be finite or infinite. Such a set is easily extended to a relation between arbitrary simple nets and nets and then to arbitrary nets and nets:

Structural reduction.



We use \sim_s for the symmetric and transitive closure of \rightsquigarrow_s .

Remark 3.1. One can check that we have provided reduction rules for all redexes compatible with our typing system: for any simple net s made of two cells connected through their principal ports, there is a reduction rule whose left member is s . This rule is unique, up to the choice of a set of labels, but this choice has no influence on the right member of the rule.

3.3.2. Confluence.

Theorem 3.2. *Let $R, R' \subseteq \mathcal{L}_\tau$. Let $\mathcal{R} \subseteq \Delta \times \mathbb{N}(\Delta)$ be the union of some of the reduction relations $\rightsquigarrow_{c,R}$, $\rightsquigarrow_{nd,R'}$, \rightsquigarrow_m and \rightsquigarrow_s . The relation \mathcal{R}^* is confluent on $\mathbb{N}(\Delta)$. \square*

The proof is essentially trivial since the rewriting relation has no critical pair (see [ER06]). Given $R \subseteq \mathcal{L}_\tau$, we consider in particular the following R -reduction: $\rightsquigarrow_R = \rightsquigarrow_m \cup \rightsquigarrow_{c,\{\tau\}} \cup \rightsquigarrow_s \cup \rightsquigarrow_{nd,R}$. We set $\rightsquigarrow_d = \rightsquigarrow_\emptyset$ (“d” for “deterministic”) and denote by \sim_d the symmetric and transitive closure of this relation. Observe that, if s and S are nets with s simple and if $s \rightsquigarrow_d S$, then S is also simple.

Some of the reduction rules we have defined depend on a set of labels. This dependence is clearly monotone in the sense that the relation becomes larger when the set of labels increases.

The reduction of nets is not normalizing mainly because of the last structural reduction rule. Additional comments can be found in [ER06].

3.3.3. A transition system of simple nets. Let l and m be distinct elements of \mathcal{L} . We call (l, m) -communication redex a communication redex whose codereliction cell is labeled by l and whose dereliction cell is labeled by m .

Definition 3.3 (The transition system $\mathbb{D}_{\mathcal{L}}$). We define a labeled transition system $\mathbb{D}_{\mathcal{L}}$ whose objects are simple nets, and transitions are labeled by pairs of distinct elements of \mathcal{L} . Let s and t be simple nets, we have $s \xrightarrow{\overline{lm}} t$ if the following holds: $s \rightsquigarrow_{\{l,m\}}^* s_0 + S$ where s_0 is a simple net which contains an (l, m) -communication redex and becomes t when one reduces this redex.

Remark 3.4. The non-deterministic steps allowed in the reduction from s to $s_0 + S$ can only involve the codereliction and dereliction labeled by l and m respectively. The communication steps only involve τ -labeled derelictions and coderelictions. In the solos calculus, solos communicate in one step through a parallel composition. This single step becomes here a

sequence of many elementary steps and our restriction allows us to avoid considering the steps which have nothing to do with the communication we are interested in.

The net S contains other possible communications, so it corresponds to other branches of non-deterministic choices.

In this paper, to compare transition systems, we only consider *strong* bisimulations [Par81] as given by the following definition.

Definition 3.5 (Bisimulation). Given two labeled transition systems $LTS_1 = (S_1, T_1)$ (with $T_1 \subseteq S_1 \times L \times S_1$) and $LTS_2 = (S_2, T_2)$ (with $T_2 \subseteq S_2 \times L \times S_2$) on the same set L of labels, a relation R between S_1 and S_2 is a *bisimulation* between LTS_1 and LTS_2 if:

- for any $(s_1, s_2) \in R$ and $(s_1, l, s'_1) \in T_1$, there exists $s'_2 \in S_2$ such that $(s_2, l, s'_2) \in T_2$;
- and for any $(s_1, s_2) \in R$ and $(s_2, l, s'_2) \in T_2$, there exists $s'_1 \in S_1$ such that $(s_1, l, s'_1) \in T_1$.

A useful particular case is $LTS_1 = LTS_2$.

Lemma 3.6. *The relation $\sim_d \subseteq \Delta \times \Delta$ is a bisimulation on $\mathbb{D}_{\mathcal{L}}$.* □

3.4. A toolbox for process calculi interpretation. We introduce now a few families of simple nets, which are built using the previously introduced basic cells. They will be used as basic modules for interpreting processes. All of these nets, but the communication areas, can be considered as *compound cells*: in reduction, they behave in the same way as cells of interaction nets.

One could have directly defined these compound cells as primitive constructions for the representation of processes. However we prefer to stress the possibility of implementing these compound objects with more basic ones coming from the logically founded theory of differential linear logic, in order to emphasize the possible application of linear logic tools to the image of our encoding.

Communication areas cannot be considered as cells in our setting since they would require more than one principal port. Interaction nets with many principal ports have been studied (in particular for representing concurrent behaviours [Maz05]) but their theory is quite different from the “one principal port” case.

3.4.1. Compound cells.

Generalized contraction and cocontraction. A *generalized contraction cell* is a simple net t . One of its free ports (which is not connected to an auxiliary port of its cells) is called the principal port of t . The other free ports (which are finitely many and are not connected to principal ports of cells of the generalized cell) are called the auxiliary ports of t . These generalized contraction cells are inductively defined:

- a wire with type $?i$ is a generalized contraction cell (we select the port with type $?i$ as the principal port of the generalized cell);
- a weakening cell gives a generalized contraction cell by connecting a wire to its principal port (the only free port of the obtained net is the principal port of the generalized cell);
- given two generalized contraction cells, by connecting the two auxiliary ports of an additional contraction cell to the principal ports of the generalized cells and by connecting the principal port of this new contraction cell to a free port, one obtains a generalized contraction cell (the free port connected to the principal port of the added contraction cell is the principal port of the generalized cell).

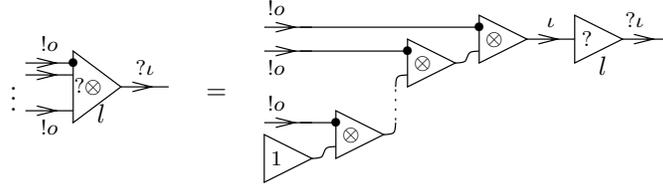


Figure 3: Dereliction-tensor compound cell.

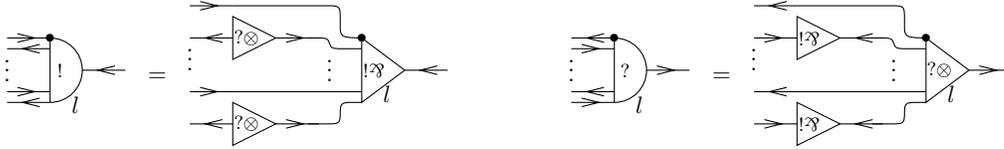


Figure 4: Input and output compound cells.

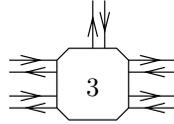


Figure 5: Area of order 3.

Generalized cocontraction cells are defined dually.

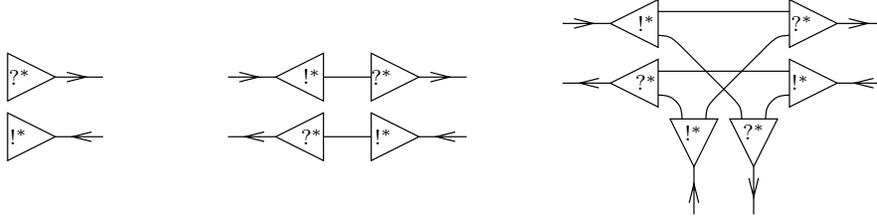
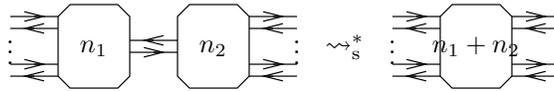
We use the same graphical notations for generalized (co)contraction cells as for ordinary (co)contraction cells, with a “*” in superscript to the “!” or “?” symbols to avoid confusions. Observe that there are infinitely many generalized (co)contraction cells of any given arity. Even if one could define a choice of one *canonical* (co)contraction cell for each arity, plugging two canonical (co)contraction cells together would not always give us a canonical one. This is why we do not define such a choice here.

The dereliction-tensor and the codereliction-par cells. Let n be a non-negative integer. We define an n -ary $?⊗$ compound cell as in Figure 3. It will be decorated by the label of its dereliction cell (as mentioned in Section 3.2, we put no decoration if the label is τ). The number of tensor cells in this compound cell is equal to n (in particular the 0-ary dereliction-tensor cell contains exactly two cells: a one cell and a dereliction cell). We define dually the $!⋈$ compound cell.

The prefix cells. Now we can define the compound cells which will play the main role in the interpretation of solos. Thanks to the above defined cells, all the oriented wires of the nets we shall define will have type $?i$ or $!o$. Therefore, we adopt the following graphical convention: the wires will have an orientation corresponding to the $?i$ type.

Let n be a non-negative integer. The n -ary *input cell* and the n -ary *output cell* are defined in Figure 4, they have n pairs of auxiliary ports $(\delta_1^+, \delta_1^-, \dots, \delta_n^+, \delta_n^-)$.

The label of a prefix cells is the one carried by its outermost $?⊗$ or $!⋈$ compound cell, the other $?⊗$ or $!⋈$ compound cells are required to be unlabeled (that is, labeled by τ).

Figure 6: Communication areas of order -1 , 0 and 1 .Figure 7: Aggregation, with $n_1, n_2 \geq -1$.

3.4.2. *Communication areas.* Let $n \geq -2$. We define a family of nets with $2(n+2)$ free ports, called *communication areas of order n* , that we shall draw using rectangles with beveled angles. Figure 5 shows how we picture a communication area of order 3.

A communication area of order n is made of $n+2$ pairs of $(n+1)$ -ary generalized cocontraction and contraction cells $(\gamma_1^+, \gamma_1^-), \dots, (\gamma_{n+2}^+, \gamma_{n+2}^-)$, with, for each i and j such that $1 \leq i < j \leq n+2$, a wire from an auxiliary port of γ_i^+ to an auxiliary port of γ_j^- and a wire from an auxiliary port of γ_i^- to an auxiliary port of γ_j^+ . We note p_i^+ and p_i^- the principal ports of γ_i^+ and γ_i^- , and we call (p_i^+, p_i^-) a pair of *associated ports* of the communication area.

So the communication area of order -2 is the empty net ϵ , and communication areas of order -1 , 0 and 1 are the structures shown in Figure 6.

3.4.3. *Useful reductions.* One of the nice properties of communication areas is that, when one connects two such areas through a pair of wires, one gets another communication area.

Lemma 3.7 (Aggregation of communication areas). *Let C_1 be a communication area of order $n_1 \geq -1$ and C_2 be a communication area of order $n_2 \geq -1$, let (p_1^+, p_1^-) be a pair of associated ports of C_1 and (p_2^+, p_2^-) be a pair of associated ports of C_2 , let s be the simple net obtained by connecting p_1^+ to p_2^- and p_1^- to p_2^+ , we have $s \rightsquigarrow_s^* C$ where C is a communication area of order $n_1 + n_2$ (see Figure 7).*

Proof. We only prove the particular case where $n_1 = n_2 = 1$ and the communication areas C_1 and C_2 are built with contraction cells (not generalized ones).

By connecting C_1 and C_2 , and by applying two structural reduction steps, we obtain a communication area of order 2 (see Figure 8). \square

Let t be a simple net and p be a port of t . We say that p is *forwarded* in t if there is a free port q of t such that t is of one of the two shapes given in Figure 9.

This allows us to describe the interaction of derelictions and coderelictions with communication areas: derelictions and coderelictions can meet each other, when connected to a common communication area.

Lemma 3.8 (Communication and forwarding of derelictions and coderelictions in communication areas). *Let $p \geq -2$ be an integer, let $l, m \in \mathcal{L}_\tau$, and let us consider the simple net*

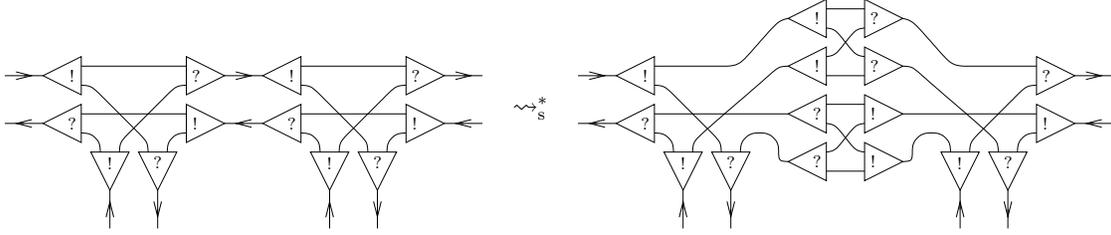


Figure 8: Aggregation of communication areas of order 1.

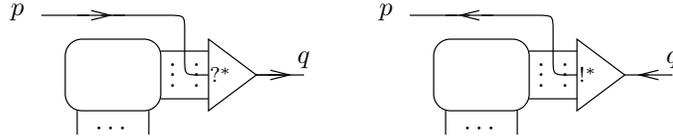


Figure 9: Port forwarding.

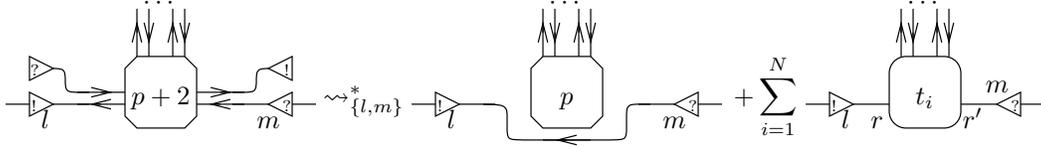


Figure 10: Dereliction and codereliction communicating through a communication area.

s obtained by connecting (the principal ports of) a codereliction labeled l and a weakening to a pair of associated ports of a communication area of order $p+2$, and by connecting (the principal ports of) a dereliction labeled m and a coweakening to another pair of associated ports of the same communication area. The $\{l, m\}$ -reduction applied to s leads to a sum of simple nets: one contains the codereliction l and the dereliction m connected through their principal ports, together with a communication area of order p ; in all the other summands, the principal ports r and r' of the codereliction l and of the dereliction m are forwarded (see Figure 10).

Proof. We consider the particular case where $p = -1$ and the communication area is built with contraction cells (not generalized ones). The reduction is pictured in Figure 11.

We first apply all the possible structural reduction steps, then we focus on the codereliction labeled l : the reduction of the associated redex gives a sum of two simple nets (in one of them the principal port of the codereliction l is forwarded). We now apply in each simple net the reduction step involving the dereliction labeled m and we finally obtain a sum of two simple nets. In the first one the principal ports of the dereliction and of the codereliction are forwarded, in the second one they are connected to each other and the remaining part of the net is a communication area of order -1 .

The generalization to other communication areas of order 1 is easy. We can obtain the case $p \neq -1$ by decomposing the communication area into two communication areas (one of order 1 and one of order $p+1$). Lemma 3.7 and Theorem 3.2 help to conclude. \square

We now consider the interaction of two prefix cells.

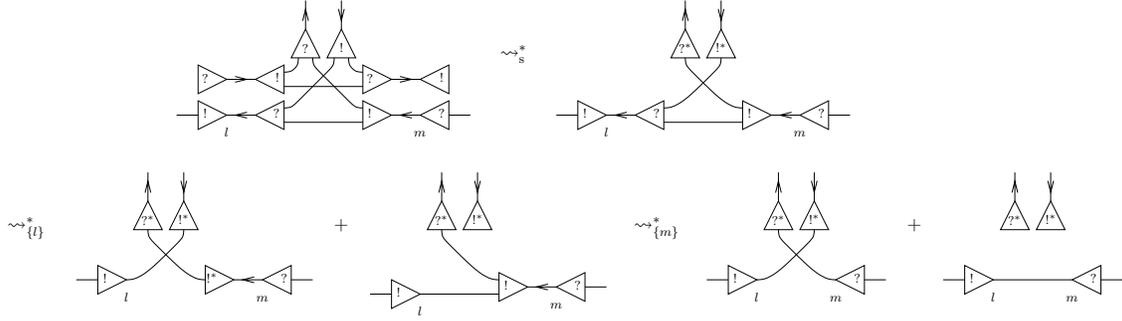


Figure 11: Forwarding of dereliction and codereliction.

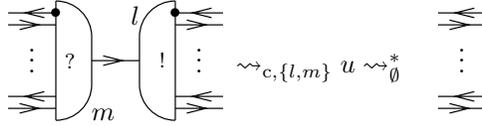


Figure 12: Prefix reduction.

Lemma 3.9 (Reduction of prefixes). *Let $l, m \in \mathcal{L}_\tau$. If we connect an n -ary output prefix labeled by m to a p -ary input prefix labeled by l through their principal ports, we obtain a simple net which reduces by $\rightsquigarrow_{c,\{l,m\}}$ to a net u which reduces by $\rightsquigarrow_{\{\tau\}}^*$ to 0 if $n \neq p$ and to simple wires by $\rightsquigarrow_{\emptyset}^*$, as in Figure 12, if $n = p$.*

Proof. The key point is to check that the simple net, obtained by connecting an n -ary dereliction-tensor compound cell to a p -ary codereliction-par cell through their principal ports, reduces to $\min(n, p)$ wires with $n - p$ weakening cells if $n > p$ or $p - n$ coweakening cells if $p > n$. \square

4. FROM SOLO DIAGRAMS TO DIFFERENTIAL INTERACTION NETS

4.1. The translation. Relying on the toolbox described above, we define a translation of labeled solo diagrams with identifications into labeled simple differential interaction nets:

- A node which appears n times as a source of a multiedge, which is k times target of a multiedge and which is p times member of an identification edge is translated as a *communication area* of order $n + k + p - 2$.

For example, for a node which is target of one input multiedge and of one output multiedge (thus $k = 2$), which is source of one multiedge (thus $n = 1$), and which is member of one identification edge (thus $p = 1$), we have $n + k + p - 2 = 2$:



- An input multiedge with sources $[x_1, x_2, x_3]$ and target \mathbf{u} is translated as a 3-ary *input cell* with principal port u^0 and auxiliary ports $[x_1^0, x_1^1, x_2^0, x_2^1, x_3^0, x_3^1]$. Each pair x_i^0, x_i^1 is connected to a pair of associated ports of the communication area corresponding to x_i .

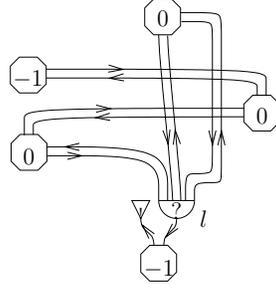
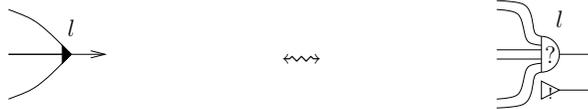


Figure 13: The translation of the second solo diagram of Figure 2.

u^0 is connected to the port p^- and a weakening cell is connected to p^+ , where (p^+, p^-) is a pair of associated ports of the communication area corresponding to \mathbf{u} . The label of the prefix cell is the label of the multiedge.



- An output multiedge with sources $[x_1, x_2, x_3]$ and target \mathbf{u} is translated as a 3-ary *output cell* with principal port u^0 and auxiliary ports $[x_1^0, x_1^1, x_2^0, x_2^1, x_3^0, x_3^1]$. Each pair x_i^0, x_i^1 is connected to a pair of associated ports of the communication area corresponding to x_i . u^0 is connected to the port p^+ and a coweakening cell is connected to p^- , where (p^+, p^-) is a pair of associated ports of the communication area corresponding to \mathbf{u} . The label of the prefix cell is the label of the multiedge.



- An identification edge connecting x_1 and x_2 is translated as a pair of wires connecting a pair (p_1^+, p_1^-) of associated ports of the communication area corresponding to x_1 with a pair (p_2^-, p_2^+) of associated ports of the communication area corresponding to x_2 .



Since communication areas of order n are not uniquely defined (because generalized contraction and cocontraction cells of a given arity are not unique either), this translation from solo diagrams to differential interaction nets is in fact a relation, which we denote $G \leftrightarrow s$ (with G a solo diagram and s a simple differential interaction net).

Remark 4.1. Let G be a solo diagram (without identification edges) and s be a differential interaction net, if $G \leftrightarrow s$ then s has no \rightsquigarrow_d redex, except some \rightsquigarrow_s redexes involving weakening or coweakening cells. Indeed, all the redexes in s are given by a labeled codereliction cell (*i.e.* whose label is not τ) or a coweakening cell facing a weakening cell, a contraction cell or a labeled dereliction cell (or dually by a dereliction cell or a weakening cell facing a coweakening or ...).

A differential interaction net associated with the second solo diagram with identifications of Figure 2 is given in Figure 13.

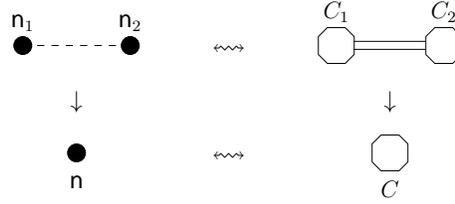


Figure 14: Proof of Lemma 4.2.

4.2. A bisimulation. Our goal is to establish a bisimulation between the labeled transition systems $\mathbb{S}_{\mathcal{L}}$ (Section 2.3) and $\mathbb{D}_{\mathcal{L}}$ (Section 3.3.3). Unfortunately the translation \rightsquigarrow does not provide such a bisimulation. We are going to show what the problems are and how to restrict $\mathbb{S}_{\mathcal{L}}$ to get a bisimulation.

4.2.1. Mismatch. The crucial point comes from step (3) of the reduction of solo diagrams: the contraction of identification edges in solo diagrams corresponds through \rightsquigarrow to the aggregation of communication areas as given by the following lemma.

Lemma 4.2. *If G_1 and G_2 are solo diagrams with identifications, if G_2 is obtained from G_1 by contracting an identification edge (step (3)) connecting the nodes \mathfrak{n}_1 and \mathfrak{n}_2 with $\mathfrak{n}_1 \neq \mathfrak{n}_2$ and if $G_1 \rightsquigarrow s_1$ then $G_2 \rightsquigarrow s_2$ where s_2 is obtained from s_1 by aggregating the communication areas corresponding to \mathfrak{n}_1 and \mathfrak{n}_2 .*

Proof. See Figure 14: G_2 is obtained from G_1 by replacing the two distinct nodes \mathfrak{n}_1 and \mathfrak{n}_2 by a unique node \mathfrak{n} . In s_1 , we have a communication area C_1 corresponding to \mathfrak{n}_1 and a communication area C_2 corresponding to \mathfrak{n}_2 . Since \mathfrak{n}_1 and \mathfrak{n}_2 are connected with an identification edge, C_1 and C_2 are connected with a pair of wires. By aggregating C_1 and C_2 (see Lemma 3.7), we obtain a communication area C in s_2 which exactly corresponds to \mathfrak{n} through \rightsquigarrow . \square

The hypothesis $\mathfrak{n}_1 \neq \mathfrak{n}_2$ is crucial since two communication areas C_1 and C_2 connected together with a pair of wires reduce to one communication area (Lemma 3.7), *except* if $C_1 = C_2$! We are thus able to encode the reduction of the solos calculus only if we can ensure that we never have to contract an identification edge connecting a node with itself.

A typical example of this problem is given by:

$$(u)(x)(y)(z)(y')(z')(a)(b)(c)(a')(b')(c')(\bar{u}xyz \mid uxy'z' \mid \bar{x}abc \mid xa'b'c')$$

As shown in Figure 15, while in solo diagrams the loop created after the first step eventually disappears, it remains in differential interaction nets. The consequence being that we have transitions in $\mathbb{S}_{\mathcal{L}}$ while the last differential interaction net has no transition to a differential interaction net which is the translation of a solo diagram.

4.2.2. Restriction. Since our goal is to make the bisimulation result true, we are going to constrain the syntax of solo diagrams in such a way that the hypotheses of Lemma 4.2 are always valid. We want to restrict the reduction step (3) to the case where the edge is not connecting a node with itself. This is equivalent to asking the sub-graph $G_{\text{id}}[e_1, e_2]$ of $G[e_1, e_2]$, containing the identification edges only, to be acyclic. By definition, we call

acyclic reduction step a reduction step associated with two dual multiedges e_1 and e_2 such that $G_{\text{id}}[e_1, e_2]$ is acyclic.

Another problem comes from the constraint on the freeness of names in the contraction of identification edges in solo diagrams. Our translation into differential interaction nets is forgetting the fact that some nodes are free and others are bound. As a consequence a reduction might happen in the differential interaction net's side without being possible in the solo diagram's side. An example is given by the term $\bar{u}xyz \mid ux'y'z'$. To avoid this situation we introduce the notion of *acyclic redex*.

Definition 4.3 (Acyclic redex). In a solo diagram, a pair of dual multiedges is an *acyclic redex* if it is a redex (meaning that the induced freeness conditions are satisfied) and the induced reduction step is an acyclic reduction step.

Definition 4.4 ($\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ labeled transition system). $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ is the biggest labeled transition system such that:

- each object of $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ is an object of $\mathbb{S}_{\mathcal{L}}$ and each transition of $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ is a transition of $\mathbb{S}_{\mathcal{L}}$ (but some objects and transitions are lost),
- for any object in $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$, all the transitions starting from it in $\mathbb{S}_{\mathcal{L}}$ belong to $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$,
- all pairs of dual multiedges in objects of $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ are acyclic redexes.

This means that an object G in $\mathbb{S}_{\mathcal{L}}$ belongs to $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ as soon as none of the paths starting from G in $\mathbb{S}_{\mathcal{L}}$ allows us to reach an object with a non-acyclic redex. This is a decidable property since there are finitely many such paths.

Definition 4.5 (Acyclic solo diagram). A solo diagram which is an object of $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ is called an *acyclic solo diagram*.

Proposition 4.6. *If $G \xrightarrow{\overline{lm}} H$ in $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ and $G \rightsquigarrow s$, then there are simple nets t and t_0 such that $s \xrightarrow{\overline{lm}} t_0$ in $\mathbb{D}_{\mathcal{L}}$, $H \rightsquigarrow t$ and $t_0 \rightsquigarrow_{\text{d}} t$.*

Proof. By hypothesis, there is a communication area C in s with two dual prefixes labeled with l and m connected to it. We apply the forwarding of derelictions and coderelictions in the communication area C to them (Lemma 3.8). This gives us a sum of simple nets containing a simple net s_1 with an (l, m) -communication redex and other summands where l and m have been forwarded.

Following Figure 16, let t_0 be the simple net obtained by reducing the (l, m) -communication redex of s_1 , and t_1 be the simple net obtained by applying to s_1 the prefix reduction starting with the reduction from s_1 to t_0 (see Lemma 3.9). It is easy to see that $G[e_1, e_2] \rightsquigarrow t_1$ where e_1 and e_2 are the multiedges of G with labels l and m . By Lemma 4.2, each step of acyclic contraction of an identification edge corresponds to the aggregation of the two associated communication areas, thus $H \rightsquigarrow t$ with $t_1 \rightsquigarrow_{\text{d}}^* t$ and we have $t_0 \rightsquigarrow_{\text{d}}^* t$. \square

Lemma 4.7 (Diving). *Let G be a solo diagram (without identification), if $G \rightsquigarrow s$ and $s \rightsquigarrow_{\{l, m\}}^* t + T$ with t containing an (l, m) -communication redex, then the codereliction labeled l and the dereliction labeled m are connected to the same communication area C in s and the (l, m) -communication redex is generated by the dereliction/codereliction forwarding through C (Lemma 3.8).*

Proof. According to Remark 4.1, the only $\rightsquigarrow_{\{l, m\}}$ redexes of s are involving the codereliction l , the dereliction m , a weakening or a coweakening.

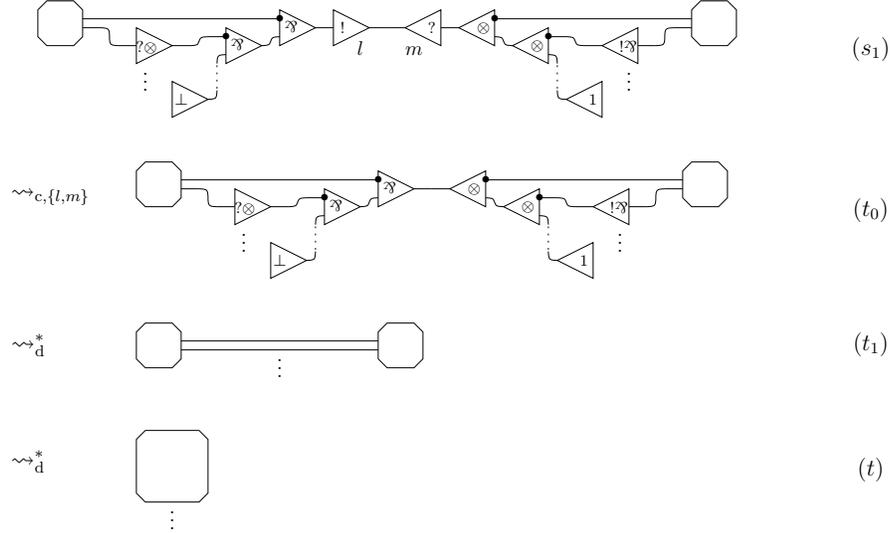


Figure 16: Simulation of a solos reduction step.

By confluence (Theorem 3.2), we can assume the reductions involving the codereliction l are coming before those involving m (and those involving the weakenening and coweakening cells arrive after them). We look at the port connected to the principal port of the codereliction l :

- (l1) If it is the auxiliary port of a prefix cell, there is no reduction for l , and no $\{l, m\}$ -reduction will change the situation (or this prefix cell is labeled m and changing the situation means erasing m).
- (l2) If it is the auxiliary port of a generalized cocontraction cell (which is not a wire), there is no reduction for l .
- (l3) If it is the principal port of a prefix cell labeled k ($k \neq m$), there is no $\{l, m\}$ -reduction for l , and no $\{l, m\}$ -reduction will change the situation.
- (l4) If it is the principal port of the prefix cell labeled m , we immediately have the result, and C is a communication area of order 0 (with the part connecting l and m which is just a simple wire).
- (l5) If it is the principal port of a generalized contraction cell of arity 0, reductions involving l will eventually erase it and we will never reach an (l, m) -communication redex, this contradicts the hypotheses.
- (l6) If it is the principal port of a generalized contraction cell of arity $n > 0$, in order to reach an (l, m) -communication redex, the reductions involving l must start destroying the generalized contraction. In order to eventually reach an (l, m) -communication redex, we must have a simple net corresponding to case 6 again, or to case 2 or 4.

We now consider these last two configurations: 6 cases followed by 2 (see Figure 18) or 4 (see Figure 17). Note that we cannot have infinitely many consecutive 6 cases since the induced reduction makes the size of the net decrease. If after 6 cases we reach case 4, l and m where connected to the same communication area C and we have the result (6 cases give a forwarding of l through C). If after 6 cases we reach case 2, we turn our attention to the reduction steps involving m . We look at the port connected to the principal port of the dereliction m :

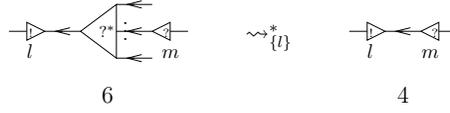


Figure 17: Proof of Lemma 4.7: sequence 6*-4.

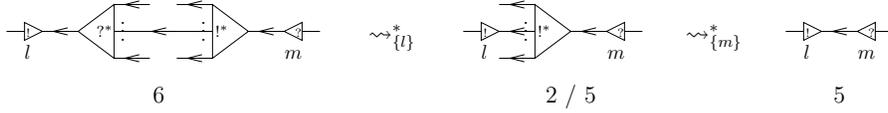


Figure 18: Proof of Lemma 4.7: sequence 6*-2-5+.

- (m1) If it is the auxiliary port of a prefix cell, there is no reduction for m , and no $\{l, m\}$ -reduction will change the situation (or this prefix cell is labeled l and changing the situation means erasing l).
- (m2) If it is the auxiliary port of a generalized contraction cell (which is not a wire), there is no reduction for m (there could be reductions involving a coweakening but which would eventually erase m), and we do not reach an (l, m) -communication redex, this contradicts the hypotheses.
- (m3) If it is the principal port of a prefix cell labeled k ($k \neq l$), there is no $\{l, m\}$ -reduction for m , and we do not reach an (l, m) -communication redex, this contradicts the hypotheses.
- (m4) If it is the principal port of a generalized cocontraction cell of arity 0, reductions involving m will eventually erase it and we will never reach an (l, m) -communication redex, this contradicts the hypotheses.
- (m5) If it is the principal port of a generalized cocontraction cell of arity $n > 0$, in order to reach an (l, m) -communication redex, the reductions involving m must start destroying the generalized cocontraction. In order to reach an (l, m) -communication redex we must escape the four cases above. This means that we go to case 5 again until we reach the simplest 5 case: m facing l (note that we cannot have infinitely many consecutive 5 cases since the size of the net decreases).

We conclude that, in order to reach an (l, m) -communication redex, we must go through a sequence of configurations 6*-4 (see Figure 17) or 6*-2-5+ (see Figure 18). For l and m to face each other after such a sequence, they must be connected to the same communication area C in s and the reduction sequence corresponds to the forwarding of l and m through C . \square

Proposition 4.8. *If $s \xrightarrow{\overline{lm}} t_0$ in $\mathbb{D}_{\mathcal{L}}$ and $G \leftrightarrow s$ with $G \in \mathbb{S}_{\mathcal{L}}^{\text{ac}}$, then there are a solo diagram H and a simple net t such that $G \xrightarrow{\overline{lm}} H$ in $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$, $H \leftrightarrow t$ and $t_0 \rightsquigarrow_{\text{d}} t$.*

Proof. Notations coincide with Figure 16. By definition of $s \xrightarrow{\overline{lm}} t_0$, there exist a simple net s_1 containing an (l, m) -communication redex and a net S such that $s \rightsquigarrow_{\{l, m\}}^* s_1 + S$ and t_0 is obtained from s_1 by reducing this (l, m) -communication redex. By Lemma 4.7, G contains two dual multiedges e_1 and e_2 connected to the same node and labeled l and m . Let t_1 be the simple net obtained by applying to s_1 the prefix reduction starting with the reduction from s_1 to t_0 (see Lemma 3.9), we first show that $G[e_1, e_2] \leftrightarrow t_1$. By Lemma 4.7 again, the only way the (l, m) -communication redex can have been generated in s_1 is by the

forwarding of the codereliction l and the dereliction m through the communication area (of order $p+2$) they are both connected to. This gives a prefix redex and a communication area of order p (Lemma 3.8). By firing this prefix redex, we generate pairs of wires connecting communication areas which exactly correspond to the identification edges of $G[e_1, e_2]$.

Let H be the solo diagram (without identifications) obtained by firing the redex between e_1 and e_2 in G (this is possible since it is an acyclic redex). We iteratively apply Lemma 4.2 starting from $G[e_1, e_2]$ and t_1 until we reach H . This is possible since $G \in \mathbb{S}_{\mathcal{L}}^{\text{ac}}$. Let t be the simple net obtained this way and corresponding to H , we have $t_0 \rightsquigarrow_d t$. \square

To get our bisimulation result, we define \rightsquigarrow_d as the composition of \rightsquigarrow and \sim_d : $G \rightsquigarrow_d s$ if there exists a simple net s_0 such that $G \rightsquigarrow s_0$ and $s_0 \sim_d s$.

Theorem 4.9 (Bisimulation). *\rightsquigarrow_d is a bisimulation between $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$ and $\mathbb{D}_{\mathcal{L}}$.*

Proof. If $G \xrightarrow{l\bar{m}} H$ in $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$, $G \rightsquigarrow s_0$ and $s_0 \sim_d s$, by Proposition 4.6 there are simple nets t and t_0 such that $s_0 \xrightarrow{l\bar{m}} t_0$ in $\mathbb{D}_{\mathcal{L}}$, $H \rightsquigarrow t$ and $t_0 \rightsquigarrow_d t$. By Lemma 3.6, there exists a simple net t_1 such that $s \xrightarrow{l\bar{m}} t_1$ and $t_0 \sim_d t_1$. As a consequence $t \sim_d t_1$ thus $H \rightsquigarrow_d t_1$.

Conversely, if $s \xrightarrow{l\bar{m}} t_1$ in $\mathbb{D}_{\mathcal{L}}$, $G \rightsquigarrow s_0$ and $s_0 \sim_d s$, by Lemma 3.6 there exists a simple net t_0 such that $s_0 \xrightarrow{l\bar{m}} t_0$ and $t_1 \sim_d t_0$. By Proposition 4.8, there are a solo diagram H and a simple net t such that $G \xrightarrow{l\bar{m}} H$ in $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$, $H \rightsquigarrow t$ and $t_0 \rightsquigarrow_d t$. As a consequence $t \sim_d t_1$ thus $H \rightsquigarrow_d t_1$.

$$\begin{array}{ccccc}
 G & \rightsquigarrow & s_0 & \sim_d & s \\
 \downarrow l\bar{m} & & \downarrow l\bar{m} & & \downarrow l\bar{m} \\
 & & t_0 & \sim_d & t_1 \\
 & & \downarrow \wr_d & & \\
 H & \rightsquigarrow & t & &
 \end{array}$$

\square

The rest of the paper will be devoted to the definition of a sub-calculus of the solos calculus whose translation into solo diagrams lives inside $\mathbb{S}_{\mathcal{L}}^{\text{ac}}$.

5. THE ACYCLIC SOLOS CALCULUS

By following the approach of the previous section and by analyzing the translation of the π -calculus into the solos calculus, we define a sub-system of the solos calculus called the *acyclic solos calculus*. The key (informal) properties of this calculus are:

- *expressiveness*: it contains the image of the π -calculus through the translation into solos (Sections 5.1.2 and 5.2.2).
- *stability*: it is well-defined with respect to the reduction of solos since it is stable under reduction (Propositions 5.4 and 5.10).
- *acyclicity*: the solo diagram associated with a term of the acyclic solos calculus is an acyclic solo diagram (and thus the bisimulation with differential interaction nets holds!) (Section 5.3).

The definition of the acyclic solos calculus is given in two steps: first by means of a typing system assigning Send/Receive polarities to occurrences of names, second by structural constraints on typed terms. Intuitively, we restrict terms to a forest-like structure

with reduction occurring only between roots of the trees of the forest (in fact we will deal with more general structures than forests). This is an abstraction of the forest structure induced by the sequentiality of a π -term on its translation into solos. The Send/Receive polarities on their side represent the input/output asymmetry of the π -calculus in the solos setting.

5.1. Types. We consider a system with only two types V and W . We use U for denoting either V or W .

A *typed term* is a term with scopes decorated with types: $(x^U)P$.

A typing judgment is of the shape $\Gamma \vdash P$ where Γ contains typing declarations associating types with names and P is a typed term. The typing rules are:

$$\frac{}{\Gamma \vdash 0} \qquad \frac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \mid Q} \qquad \frac{\Gamma, x : U \vdash P}{\Gamma \vdash (x^U)P}$$

$$\frac{}{\Gamma, x : V, y : V, z : W, z' : W \vdash \hat{x} z z' y}$$

$$\frac{}{\Gamma, x : W, z : W, y : V, y' : V \vdash \hat{x} z y y'}$$

where \hat{x} is either x or \bar{x} .

An intuitive way of understanding the types V and W is given by these two mutually recursive definitions:

$$V ::= WWW$$

$$W ::= WVW$$

(more generally, starting from other mutually recursive definitions of a finite set of types, one could derive other typing systems of the same kind which would also satisfy the subject reduction property for example).

The main purpose of this typing system is to be able to associate a *communication protocol* with each object occurrence. Communication protocols are **S** (send) and **R** (receive). If we add communication protocols as a decoration on the definitions of V and W :

$$V ::= W^R W^S V^S$$

$$W ::= W^R V^S V^R$$

then any object occurrence of a typed term in context can be decorated with a communication protocol: in an input solo whose subject has type U , the decoration is given according to those of the components of the type, in an output solo it is given in a dual way: $\mathbf{S} \mapsto \mathbf{R}$ and $\mathbf{R} \mapsto \mathbf{S}$. More formally, we can directly enrich typing derivations in such a way that they assign communication protocols to object occurrences of names in a typed term:

$$\frac{}{\Gamma \vdash 0} \qquad \frac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \mid Q} \qquad \frac{\Gamma, x : U \vdash P}{\Gamma \vdash (x^U)P}$$

$$\frac{}{\Gamma, x : V, y : V, z : W, z' : W \vdash x z^R z'^S y^S}$$

$$\frac{}{\Gamma, x : V, y : V, z : W, z' : W \vdash \bar{x} z^S z'^R y^R}$$

$$\frac{}{\Gamma, x : W, z : W, y : V, y' : V \vdash x z^R y^S y'^R}$$

$$\frac{}{\Gamma, x : W, z : W, y : V, y' : V \vdash \bar{x} z^S y^R y'^S}$$

Examples are given by the image of the translation of the π -calculus in Section 5.1.2.

The intuition behind **S** and **R** comes from the difference between the name-passing mechanism in the π -calculus and in the solos calculus. Name-passing by substitution (as in π) entails that a flow of information is given by interaction from an output prefix to an input prefix (receiving names are substituted with sending names). Name-passing by unification (as in fusion or solos) establishes a perfect symmetry between the agents trying to communicate. Communication protocols **S** and **R** are used to trace the flow of information coming from π after translation into solos. Even if communication is symmetric (inside the solos calculus), it is reasonable for a term decorated with communication protocols to understand the unification of x^S with y^R as y being substituted with x (this will be a consequence of the constraints given in Definition 5.6).

5.1.1. *Preservation by reduction.* We first check elementary properties of the typing system with respect to reduction.

Lemma 5.1 (Free names). *If $\Gamma \vdash P$ then all the free names of P appear in Γ .*

Lemma 5.2 (Weakening). *If $x \notin \Gamma$ and $x \notin \text{fn}(P)$, $\Gamma \vdash P$ if and only if $\Gamma, x : U \vdash P$.*

Lemma 5.3 (Substitution). *If $\Gamma, x : U, y : U \vdash P$ then $\Gamma, x : U \vdash P[x/y]$.*

Proof. These three lemmas are proved by simple inductions on the typing derivations. \square

Proposition 5.4 (Subject reduction).

- (1) *If $P \equiv Q$ then $\Gamma \vdash P \Leftrightarrow \Gamma \vdash Q$.*
- (2) *If P reduces to Q and $\Gamma \vdash P$ then $\Gamma \vdash Q$.*

Proof.

- (1) We simply have to consider each case of the definition of \equiv (given in Section 1.2). The only interesting one is $((x)P) \mid Q \equiv (x)(P \mid Q)$ with $x \notin \text{fn}(Q)$. We assume $\Gamma \vdash ((x^U)P) \mid Q$, this entails $\Gamma \vdash (x^U)P$ and $\Gamma \vdash Q$ and thus $\Gamma, x : U \vdash P$. By Lemma 5.2, we have $\Gamma, x : U \vdash Q$ thus $\Gamma, x : U \vdash P \mid Q$ and finally $\Gamma \vdash (x^U)(P \mid Q)$. Conversely, if $\Gamma \vdash (x^U)(P \mid Q)$ then $\Gamma, x : U \vdash P \mid Q$, $\Gamma, x : U \vdash P$, $\Gamma, x : U \vdash Q$ thus, by Lemma 5.2 with $x \notin \text{fn}(Q)$, $\Gamma \vdash Q$ and:

$$\frac{\frac{\Gamma, x : U \vdash P}{\Gamma \vdash (x^U)P} \quad \Gamma \vdash Q}{\Gamma \vdash ((x^U)P) \mid Q}$$

- (2) The only interesting case is $(\tilde{z})(\bar{u}x_1x_2x_3 \mid uy_1y_2y_3 \mid P) \rightarrow P\sigma$ where \tilde{z} contains exactly the names such that $\sigma(w) \neq w$. From the hypothesis, we deduce $\Gamma, \tilde{z} : \tilde{U} \vdash \bar{u}x_1x_2x_3$ and $\Gamma, \tilde{z} : \tilde{U} \vdash uy_1y_2y_3$ and $\Gamma, \tilde{z} : \tilde{U} \vdash P$. By Lemma 5.3, we have $\Gamma \vdash P\sigma$. \square

5.1.2. *Typability of the translation.* We prove the typability of the translation of any π -term (see Section 1.3 for the definition of the translation).

If P is a π -term with free names in the finite set $\mathcal{X} = \{x_1, \dots, x_k\}$, we define $\Gamma_{\mathcal{X}}$ to be the environment $x_1 : W, \dots, x_k : W$ and $\Gamma_{\mathcal{X},v}$ to be the environment $\Gamma_{\mathcal{X}}, v : V$.

Proposition 5.5 (Typability). *If P is a π -term with free names in the finite set \mathcal{X} , $\Gamma_{\mathcal{X}} \vdash [P]$.*

Proof. We first show that $\Gamma_{\mathcal{X},v} \vdash [P]_v$.

The process C_v is typable:

$$\frac{\Gamma_{\mathcal{X},v}, z : W \vdash v z^{\mathbf{R}} z^{\mathbf{S}} v^{\mathbf{S}}}{\Gamma_{\mathcal{X},v} \vdash (z^W) v z^{\mathbf{R}} z^{\mathbf{S}} v^{\mathbf{S}}}$$

The translations of prefixes are typable (see Figure 19).

The cases of $P \mid Q$ and $\nu x.P$ are immediate.

Finally we get the typability of $[P]$:

$$\frac{\frac{\Gamma_{\mathcal{X},v} \vdash [P]_v \quad \Gamma_{\mathcal{X},v} \vdash C_v}{\Gamma_{\mathcal{X},v} \vdash [P]_v \mid C_v}}{\Gamma_{\mathcal{X}} \vdash (v^V) ([P]_v \mid C_v)}$$

□

We can sum up the induced communication protocols:

$$\begin{aligned} C_v &= (z^W) v z^{\mathbf{R}} z^{\mathbf{S}} v^{\mathbf{S}} \\ [u(x).P]_v &= (w^W) (y^V) (\bar{v} u^{\mathbf{S}} w^{\mathbf{R}} y^{\mathbf{R}} \mid C_y \mid (x^W) (v'^V) (w x^{\mathbf{R}} v^{\mathbf{S}} v'^{\mathbf{R}} \mid [P]_{v'})) \\ [\bar{u}\langle x \rangle.P]_v &= (w^W) (y^V) (\bar{v} u^{\mathbf{S}} w^{\mathbf{R}} y^{\mathbf{R}} \mid C_y \mid (v'^V) (\bar{w} x^{\mathbf{S}} v'^{\mathbf{R}} v^{\mathbf{S}} \mid [P]_{v'})) \\ [P \mid Q]_v &= [P]_v \mid [Q]_v \\ [\nu x.P]_v &= (x^W) [P]_v \\ [P] &= (v^V) ([P]_v \mid C_v) \end{aligned}$$

5.2. Acyclicity. Relying on the decoration with communication protocols induced by the typing system, we are now able to define our restriction of the solos calculus.

Given a typed term P , we first define some properties and relations between names and solos occurring in P . If s is a solo, we write $\text{subj}(s)$ for its subject.

$$\begin{aligned} x \in s &:= x \text{ has an object occurrence in } s \\ x^{\mathbf{X}} \in s &:= x \text{ has an object occurrence in } s \text{ with communication protocol } \mathbf{X} \\ s \triangleleft x &:= x^{\mathbf{R}} \in s \\ s \triangleleft t &:= s \triangleleft \text{subj}(t) \\ s \perp t &:= \text{subj}(s) = \text{subj}(t), \text{ one is output and the other is input} \\ s \text{ is a root} &:= \text{there is no } t \text{ such that } t \triangleleft s \end{aligned}$$

We denote by \triangleleft^+ the transitive closure of \triangleleft and by \triangleleft^* the reflexive transitive closure of \triangleleft .

Definition 5.6 (Acyclic solos calculus). *Acyclic solos terms* are typed terms which satisfy the following five properties:

- (AC1) each name has at most one \mathbf{R} -occurrence
- (AC2) $s \perp t$ implies that s and t are roots
- (AC3) $s \triangleleft x$ and $x \in t$ implies $s \triangleleft^* t$
- (AC4) $x^{\mathbf{S}} \in s$ and $s \triangleleft x$ implies s is an input
- (AC5) there is no free name with an \mathbf{R} -occurrence

$$\begin{array}{c}
\frac{\Gamma_{\mathcal{X},v,y : V, w : W, v' : V, x : W \vdash w x^R v^S v'^R} \quad \Gamma_{\mathcal{X},v,y : V, w : W, v' : V, x : W \vdash [P]_{v'}}}{\Gamma_{\mathcal{X},v,y : V, w : W, v' : V, x : W \vdash w x^R v^S v'^R \mid [P]_{v'}}} \\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W, v' : V, x : W \vdash w x^R v^S v'^R \mid [P]_{v'}} \quad \Gamma_{\mathcal{X},v,y : V, w : W, x : W \vdash (v'^V)(w x^R v^S v'^R \mid [P]_{v'})}}{\Gamma_{\mathcal{X},v,y : V, w : W \vdash (x^W)(v'^V)(w x^R v^S v'^R \mid [P]_{v'})}} \\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W \vdash \bar{v} u^S w^R y^R \mid C_y} \quad \Gamma_{\mathcal{X},v,y : V, w : W \vdash C_y}}{\Gamma_{\mathcal{X},v,y : V, w : W \vdash \bar{v} u^S w^R y^R \mid C_y}} \\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W \vdash \bar{v} u^S w^R y^R \mid C_y} \quad (x^W)(v'^V)(w x^R v^S v'^R \mid [P]_{v'})}{\Gamma_{\mathcal{X},v,w : W \vdash (y^V) \bar{v} u^S w^R y^R \mid C_y} \quad (x^W)(v'^V)(w x^R v^S v'^R \mid [P]_{v'})} \\
\frac{\Gamma_{\mathcal{X},v,w : W \vdash (y^V) \bar{v} u^S w^R y^R \mid C_y} \quad (x^W)(v'^V)(w x^R v^S v'^R \mid [P]_{v'})}{\Gamma_{\mathcal{X},v \vdash (w^W)(y^V) \bar{v} u^S w^R y^R \mid C_y} \quad (x^W)(v'^V)(w x^R v^S v'^R \mid [P]_{v'})} \\
\\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W, v' : V \vdash \bar{w} x^S v'^R v^S} \quad \Gamma_{\mathcal{X},v,y : V, w : W, v' : V \vdash [P]_{v'}}}{\Gamma_{\mathcal{X},v,y : V, w : W, v' : V \vdash \bar{w} x^S v'^R v^S \mid [P]_{v'}}} \\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W, v' : V \vdash \bar{w} x^S v'^R v^S \mid [P]_{v'}} \quad \Gamma_{\mathcal{X},v,y : V, w : W, v' : V \vdash (v'^V)(\bar{w} x^S v'^R v^S \mid [P]_{v'})}}{\Gamma_{\mathcal{X},v,y : V, w : W \vdash (v'^V)(\bar{w} x^S v'^R v^S \mid [P]_{v'})}} \\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W \vdash \bar{v} u^S w^R y^R \mid C_y} \quad \Gamma_{\mathcal{X},v,y : V, w : W \vdash C_y}}{\Gamma_{\mathcal{X},v,y : V, w : W \vdash \bar{v} u^S w^R y^R \mid C_y}} \\
\frac{\Gamma_{\mathcal{X},v,y : V, w : W \vdash \bar{v} u^S w^R y^R \mid C_y} \quad (v'^V)(\bar{w} x^S v'^R v^S \mid [P]_{v'})}{\Gamma_{\mathcal{X},v,w : W \vdash (y^V) \bar{v} u^S w^R y^R \mid C_y} \quad (v'^V)(\bar{w} x^S v'^R v^S \mid [P]_{v'})} \\
\frac{\Gamma_{\mathcal{X},v,w : W \vdash (y^V) \bar{v} u^S w^R y^R \mid C_y} \quad (v'^V)(\bar{w} x^S v'^R v^S \mid [P]_{v'})}{\Gamma_{\mathcal{X},v \vdash (w^W)(y^V) \bar{v} u^S w^R y^R \mid C_y} \quad (v'^V)(\bar{w} x^S v'^R v^S \mid [P]_{v'})}
\end{array}$$

Figure 19: Typing derivations for the translations of prefixes.

The *acyclic solos calculus* is given by restricting terms to acyclic solos terms. The structural congruence and the reduction relation are those induced by the solos calculus.

We make a few remarks and state immediate consequences of this definition:

- (1) Let us consider the following definition of communication protocols for occurrences of names in a π -term: if the occurrence is an object occurrence in an input prefix, the communication protocol is **R**, otherwise it is **S**. Assuming that the names appearing in a π -term are made as different as possible by α -conversion, we can remark that each name has at most one **R**-occurrence, and no free name has an **R**-occurrence.
- (2) (AC2) entails that reduction only occurs between roots.
- (3) In the case where \triangleleft induces a forest ordering, (AC3) means that all the **S**-occurrences of x are bigger (with respect to this forest ordering) than its **R**-occurrence (if any).
- (4) (AC4) says that a name can have both an **S**-occurrence and an **R**-occurrence in a solo s only if s is an input. This is a place where we are breaking the symmetry of the solos calculus between output solos and input solos. This allows us to rule out processes like $x y^{\mathbf{S}} y^{\mathbf{R}} u^{\mathbf{R}} \mid \bar{x} z^{\mathbf{R}} z^{\mathbf{S}} v^{\mathbf{S}}$ that would lead to twice the identification of y with z which is almost the same as identifying y with y (and would break acyclicity of the associated solo diagram).
- (5) (AC5) ensures (with (AC1)) that identification edges are always contractible (no freeness problem).

5.2.1. *Preservation by reduction.* We want to prove that reducts of an acyclic solos term under the reduction of the solos calculus are all acyclic solos terms (Proposition 5.10). We consider a reduction from P to Q by an interaction between the solos s_0 and t_0 of P . By definition of reduction (see Section 1.2), we have a substitution σ such that each solo t of Q is of the shape $s\sigma$ for some solo s of P . We say that t is a *residue* of s . The solos s_0 and t_0 of P have no residue in Q while all the other solos of P have a unique residue in Q . In this section, in order to simplify the notations, when P reduces to Q we will use the same name s for a solo in P and for its unique residue in Q if it exists.

To make the difference between the relations \triangleleft in different processes, we use the notation \triangleleft_P for the \triangleleft relation of the process P .

The following three lemmas hold assuming only conditions (AC1), (AC2) and (AC3) on P .

Lemma 5.7. *If P is a typed term of the solos calculus satisfying (AC1), (AC2) and (AC3) and which reduces to Q , then no **R**-occurrence of Q has been substituted during reduction.*

Proof. Let s_0 and t_0 be the solos destroyed during the reduction from P to Q , if a name x involved in the reduction (*i.e.* taking part in the unification, *i.e.* occurring in s_0 or t_0) has its **R**-occurrence in a solo t of P different from s_0 and t_0 then $t \triangleleft_P x$ and, by (AC3), either $t \triangleleft_P^{\dagger} s_0$ or $t \triangleleft_P^{\dagger} t_0$ thus one of them is not a root of P contradicting (AC2). \square

Lemma 5.8. *If P is a typed term of the solos calculus satisfying (AC1), (AC2) and (AC3) and which reduces to Q , if s is a solo in P whose subject has been modified during this reduction (*i.e.* the subject of s in P is not the same as the subject of its residue in Q), we have both:*

- (the residue of) s is a root in Q
- s is a root in P or $s_0 \triangleleft_P s$ or $t_0 \triangleleft_P s$

Proof. Let x be the subject of s in P and y be its subject in Q , each of x and y appears in either s_0 or t_0 . If s is not a root in Q then there is some t such that $t \triangleleft_Q s$ thus $t \triangleleft_P y$ (by Lemma 5.7). If $y \in s_0$ (and symmetrically if $y \in t_0$) then $t \triangleleft_P^* s_0$ (by (AC3)) with $t \neq s_0$ since t belongs to Q thus s_0 is not a root in P contradicting (AC2). So that s is a root in Q .

If s is not a root in P then there is some t such that $t \triangleleft_P s$. Since the subject of s is modified, it means that an R-occurrence in t is also substituted thus $t \notin Q$ by Lemma 5.7 and $t = s_0$ or $t = t_0$. \square

Lemma 5.9. *If P is a typed term of the solos calculus satisfying (AC1), (AC2) and (AC3) and which reduces to Q , if the reduction introduces an object occurrence of the name x in Q then x has no R-occurrence in Q .*

Proof. For the reduction to put some x in some t , x must occur in s_0 or t_0 . Assume that x occurs in s_0 and that x has an R-occurrence in s in Q (so that $s \triangleleft_Q x$). By Lemma 5.7, $s \triangleleft_P x$ and by (AC3), we have $s \triangleleft_P^* s_0$ but, since $s \neq s_0$ (because $s_0 \notin Q$), this would entail that s_0 is not a root in P contradicting (AC2). \square

We now turn to the preservation result.

Proposition 5.10 (Acyclicity preservation). *If P is an acyclic solos term and P reduces to Q then Q is an acyclic solos term.*

Proof. We first prove the preservation of conditions (AC1), (AC2) and (AC3), independently of conditions (AC4) and (AC5).

Preservation of condition (AC1) is given by Lemma 5.7.

We move to condition (AC2). If $s \perp_Q s'$, note that if one of them is not a root then none of them is since $t \triangleleft s \Rightarrow t \triangleleft \text{subj}(s) \Rightarrow t \triangleleft \text{subj}(s') \Rightarrow t \triangleleft s'$. Assume that $t \triangleleft_Q s$ and $t \triangleleft_Q s'$, if the subject of neither s nor s' has been modified during reduction, then $t \triangleleft_P s$ and $t \triangleleft_P s'$ (since the R-occurrence in t of the subject of s and s' has not been modified, by Lemma 5.7) and P contradicts condition (AC2). If one of the subjects of s or s' has been modified then it is a root in Q by Lemma 5.8.

Concerning condition (AC3), we assume $s \triangleleft_Q x$ and $x \in_Q t$. By Lemma 5.7, $s \triangleleft_P x$ and by Lemma 5.9, $x \in_P t$. So that, by (AC3), $s \triangleleft_P^* t$. If $s \not\triangleleft_Q^* t$, then the path from s to t for the relation \triangleleft_P has been broken during reduction, this entails that a solo $t' \neq s$ in this path has got his subject modified. By Lemma 5.8, $s_0 \triangleleft_P t'$ or $t_0 \triangleleft_P t'$ but this is impossible because, since s_0 and t_0 are roots in P and are not in Q , $s \neq s_0$ and $s \neq t_0$.

We now assume that (AC1), (AC2) and (AC3) hold. Condition (AC4) is preserved: if, in Q , an output solo s contains both an R-occurrence and an S-occurrence of a name x then, by Lemmas 5.7 and 5.9, it is also the case in P . Condition (AC5) is preserved by Lemma 5.7. \square

All this shows that our acyclic solos calculus is well behaved with respect to the reduction of the solos calculus.

5.2.2. *Acyclicity of the translation.* In order to prove the expressiveness of the acyclic solos calculus, we check that it contains the image of the translation of the π -calculus.

By construction, edges in $G_{\text{id}}[e_1, e_2]$ are connecting two nodes which correspond to occurrences of names in the term P which are going to be unified. By definition of the acyclic solos calculus, one of these occurrences is an **S**-occurrence and the other one is an **R**-occurrence. We consider the directed graph G' obtained by orienting the edges of $G_{\text{id}}[e_1, e_2]$ towards **R**-occurrences.

We first prove the following lemma about directed and non-directed graphs.

Lemma 5.13. *Let g be a finite directed graph and g_s be the underlying non-directed graph. If g_s is connected, if g has a root (that is a node without incoming edge) and if each node of g has at most one incoming edge then g_s is acyclic.*

Proof. The function which maps each edge of g to its target is an injective function (by hypotheses) and its image is the set of nodes of g which are not roots. As a consequence, the number of edges of g (and thus of g_s) is less than or equal to the number of vertices of g (and thus of g_s) minus one. Since g_s is connected, we conclude that it is acyclic. \square

We can now prove Lemma 5.12:

Proof. We prove that the connected components of $G_{\text{id}}[e_1, e_2]$ equipped with the orientation of G' satisfy the hypotheses of Lemma 5.13 and thus that $G_{\text{id}}[e_1, e_2]$ is acyclic. Let us assume e_1 is the input multiedge in the redex between e_1 and e_2 in G (since solos in a solos term are in one-to-one correspondence with multiedges in the corresponding solo diagram, we use the notations introduced in the beginning of Section 5.2 with multiedges as well as with solos), we have the following properties:

- (i) if $x^{\text{R}} \in e_1$ then all the other occurrences of x in e_1 and e_2 are **S**-occurrences in e_1 : by (AC1), x cannot have another **R**-occurrence, moreover, if $x^{\text{S}} \in e_2$ then, by (AC3), $e_1 \triangleleft^+ e_2$ thus e_2 is not a root of P , contradicting (AC2).
- (ii) if $x^{\text{R}} \in e_2$ then x has no other occurrence neither in e_1 nor in e_2 : as for i, x cannot have any other **R**-occurrence nor any **S**-occurrence in e_1 and finally, by (AC4), it cannot have an **S**-occurrence in e_2 .
- (iii) if $x^{\text{S}} \in e_2$ then x has no **R**-occurrence neither in e_1 nor in e_2 : otherwise we apply i or ii.

We consider a connected component G_0 of $G_{\text{id}}[e_1, e_2]$. It is the underlying non-directed graph of the appropriate sub-graph G'_0 of G' . G'_0 is finite, G_0 is connected and, according to condition (AC1), each node has at most one incoming edge in G'_0 . We just have to show that G'_0 has a root, *i.e.* that there is a name in P which has only **S**-occurrences in e_1 and e_2 for each connected component G_0 . We first show that any **R**-occurrence in e_2 is connected to an **S**-occurrence of a name without **R**-occurrence: we start from this **R**-occurrence in e_2 , the corresponding occurrence in e_1 is an **S**-occurrence of a name x , if x has no **R**-occurrence we are done, and if x has an **R**-occurrence then it is also in e_1 (by ii) and the corresponding occurrence in e_2 is an **S**-occurrence (thus without **R**-occurrence by iii). Since any connected component contains either an **R**-occurrence in e_2 or an **S**-occurrence in e_2 , we are done (with iii again).

Finally by (AC5), G' has the property that only roots can be free nodes (according to the free/bound labeling of nodes in G). This property is preserved by contraction of identification edges thanks to (AC1). Moreover this property entails that at most one extremity of an identification edge can be free, and thus no freeness problem arises during contraction of identification edges. \square

Theorem 5.14 (Acyclic diagrams from acyclic terms). *The solo diagram associated with an acyclic solos term (Definition 5.6) is an acyclic solo diagram (Definition 4.5).*

Proof. We prove that the range of the translation of the acyclic solos calculus into solo diagrams satisfies the conditions of Definition 4.4. By Lemma 5.12, we know such a translation contains only acyclic redexes. Now if G is the translation of the acyclic solos term P and reduces to H , there exists a term Q which translates into H , and such that P reduces to Q thus Q is an acyclic solos term (Proposition 5.10). \square

CONCLUSION

In the spirit of a Curry-Howard correspondence between a logical device and a concurrent one, we have shown how to stress a strong connection between differential interaction nets and the solos calculus. Such links allow one to share methods from the two worlds, with the possibility of giving concurrent interpretations to the cut-elimination procedure, logical foundations to process calculi, of applying tools from concurrency theory such as behavioural equivalences to formal proofs, or conversely denotational semantics from linear logic to concurrent languages, ...

Let us discuss a few constraints or restrictions we have to face here.

Finitary calculi. We have only considered finitary calculi: without replications nor recursive definitions. Nevertheless, *exponential boxes* are a natural device from linear logic proof-nets (compatible with differential interaction nets) for representing replicable processes. We have shown in [EL10] how to represent with these boxes a restricted (but expressive) form of replication of the π -calculus. This restriction is related with difficulties for controlling reduction on the auxiliary ports of exponential boxes. By exploiting this kind of restriction in the language of solos, it should be possible to represent a restricted form of replication through our translation.

Logical correctness. Proof-nets of linear logic provide us with a graphical syntax for representing proofs of the sequent calculus [Gir96]. However this requires to impose a *correctness criterion* to proof-nets in order to characterize exactly those which could be sequentialized into a sequent calculus proof. There is such a correctness criterion for differential interaction nets [ER06] (relating them with the sequent calculus of differential linear logic). However the differential interaction nets obtained with our translation do not satisfy the correctness criterion in general (this comes in particular from the unconstrained use of communication areas). Some logical tools do not care about logical correctness and can be directly applied in our setting (for example, the relational model is a denotational model of differential interaction nets even if they do not satisfy correctness). On the contrary, many important results in logic only hold in the logically correct case. This is why it would be important to understand how our translation interacts with correctness, or how expressive would be a concurrent calculus whose translation into differential interaction nets satisfies the correctness criterion. In particular, what kind of communications could be represented with communication areas in a logically correct setting?

Acyclicity. In trying to build a strong bridge between differential interaction nets and solo diagrams, we have been able to define a suitable restriction of the solos calculus that can be used as an intermediary step between the π -calculus and differential interaction nets. This led to a new proof of the bisimulation result presented in [EL10].

The technical choices in the design of our acyclic solos calculus were completely determined by the just mentioned goal. Nevertheless, it would be interesting to study acyclic solos for themselves. Their computational behaviour are in many ways similar to what happens in the π -calculus (“substitution-like” name passing in an “unification style” setting for example). It would be interesting to see if they contain specific communication primitives or if somehow the behaviour of an acyclic solos term always mimics the behaviour of a π -term.

Another approach would be to extend our translation relation to take cycles (as in Figure 15) into account. This requires us to understand the precise impact of these cycles on the behaviour of differential interaction nets. In this way, it seems possible to extend the bisimulation result to the whole solos calculus.

Following the link provided by the present work, the above mentioned points should now be addressed to strengthen the idea of an underlying Curry-Howard correspondence. However some possibilities are already open such as relational semantics for the π -calculus, geometry of interaction for processes, ...

ACKNOWLEDGEMENT

We would like to thank Cosimo Laneve for the helpful discussions we had about solos. We also thank the anonymous referees for their useful comments.

REFERENCES

- [Abr93] Samson Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111(1–2):3–57, 1993.
- [Ale99] Vladimir Alexiev. *Non-deterministic Interaction Nets*. Ph.D. thesis, University of Alberta, 1999.
- [BM06] Emmanuel Beffara and François Maurel. Concurrent nets: a study of prefixing in process calculi. *Theoretical Computer Science*, 356(3):356–373, May 2006.
- [Bou92] Gérard Boudol. Asynchrony and the pi-calculus. Research Report 1702, INRIA, 1992.
- [BS94] Gianluigi Bellin and Philip J. Scott. On the pi-calculus and linear logic. *Theoretical Computer Science*, 135(1):11–65, 1994.
- [Ehr02] Thomas Ehrhard. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12(5):579–623, 2002.
- [EL10] Thomas Ehrhard and Olivier Laurent. Interpreting a finitary pi-calculus in differential interaction nets. *Information and Computation*, 208(6):606–633, June 2010.
- [ER06] Thomas Ehrhard and Laurent Regnier. Differential interaction nets. *Theoretical Computer Science*, 364(2):166–195, 2006.
- [FM05] Claudia Faggian and François Maurel. Ludics nets, a game model of concurrent interaction. In *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science*, pages 376–385. IEEE Computer Society, 2005.
- [Gir96] Jean-Yves Girard. Proof-nets: the parallel syntax for proof-theory. In Aldo Ursini and Paolo Agliano, editors, *Logic and Algebra*, volume 180 of *Lecture Notes In Pure and Applied Mathematics*, pages 97–124, New York, 1996. Marcel Dekker.
- [HL10] Kohei Honda and Olivier Laurent. An exact correspondence between a typed pi-calculus and polarised proof-nets. *Theoretical Computer Science*, 411(22–24):2223–2238, May 2010.

- [HT91] Kohei Honda and Mario Tokoro. An object calculus for asynchronous communication. In *Proceedings of ECOOP'91*, volume 512 of *Lecture Notes in Computer Science*, pages 133–147. Springer-Verlag, 1991.
- [HY94] Kohei Honda and Nobuko Yoshida. Combinatory representation of mobile processes. In *POPL '94: Proceedings of the 21st ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 348–360, New York, NY, USA, 1994. ACM.
- [JM04] Ole Jensen and Robin Milner. Bigraphs and mobile processes (revised). Technical report, Cambridge University Computer Laboratory, 2004.
- [Kha03] Lionel Khalil. *Généralisation des Réseaux d'Interaction avec amb, l'agent de McCarthy : propriétés et applications*. Thèse de doctorat, Ecole Normale Supérieure de Paris, June 2003.
- [Laf95] Yves Lafont. From proof nets to interaction nets. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Note Series*, pages 225–247. Cambridge University Press, 1995.
- [LPV01] Cosimo Laneve, Joachim Parrow, and Björn Victor. Solo diagrams. In *Proceedings of the 4th conference on Theoretical Aspects of Computer Science, TACS'01*, number 2215 in *Lecture Notes in Computer Science*, pages 127–144. Springer-Verlag, 2001.
- [LV03] Cosimo Laneve and Björn Victor. Solos in concert. *Mathematical Structures in Computer Science*, 13(5):657–683, 2003.
- [Maz05] Damiano Mazza. Multiport interaction nets and concurrency. In *Proceedings of CONCUR 2005*, number 3653 in *Lecture Notes in Computer Science*, pages 21–35. Springer-Verlag, 2005.
- [Maz06] Damiano Mazza. *Interaction Nets: Semantics and Concurrent Extensions*. Thèse de doctorat, Université Aix-Marseille II / Università degli Studi Roma Tre, 2006.
- [Mil94] Robin Milner. Pi-nets: A graphical form of pi-calculus. In *ESOP '94: Proceedings of the 5th European Symposium on Programming*, pages 26–42, London, UK, 1994. Springer-Verlag.
- [MPW92] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I/II. *Information and Computation*, 100(1):1–77, 1992.
- [Par81] David Park. Concurrency and automata on infinite sequences. In Peter Deussen, editor, *5th GI-conference on Theoretical Computer Science*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, March 1981.
- [PV98] Joachim Parrow and Björn Victor. The fusion calculus: Expressiveness and symmetry in mobile processes. In *Proceedings of the thirteenth annual symposium on Logic In Computer Science*, pages 176–185, Indianapolis, June 1998. IEEE, IEEE Computer Society Press.
- [Reg92] Laurent Regnier. *Lambda-Calcul et Réseaux*. Thèse de doctorat, Université Paris VII, 1992.