Concurrent Systems Analysis Using ECATNets

F. BELALA and MOHAMED BETTAZ, *Laboratoire LIRE, Université de Constantine, 25000 Constantine, Algeria, E-mail: bettaz@ist.cerist.dz*

LAURE PETRUCCI-DAUCHY, *LSV,CNRS UMR 8643,ENS de Cachan,61 Avenue du Président Wilson 94235 CACHAN Cedex, France. E-mail: petrucci@lsv.ens-cachan.fr*

Abstract

The main objective of this paper is to show how to extend the ECATNet model, which is a form of high-level algebraic nets, with new objects and morphisms in order to have a more expressive model-based diagnosis of concurrent systems. Our formulation is accomplished by exploiting the similarity between the categorical models of linear logic and those of ECATNets which are also categories in the rewriting logic framework.
The categorical interpretation of the extra structure is inspired from that of some linear logic connectors. In particular, the useful interpretation of the extra object $\perp$, provided a richer specification language, based on ECATNets, for the study of "negative" properties.

1 Introduction

During the past few years many researchers have established relationships between linear logic and Petri nets. Specifically the correspondence has been established, through different methods, between provability of formulas in linear logic and reachability of markings in Petri nets. One of the aims of these works has been to provide a richer language, based on linear logic, for the specification and the study of properties of concurrent computations. The main objective of this paper is to show how to extend the ECATNet model, which is a form of high-level algebraic nets [5], [4], with new objects and morphisms in order to have a more expressive model based diagnosis of concurrent systems. Our formulation is accomplished by exploiting the similarity between the categorical models of linear logic and those of ECATNets which are also categories in the rewriting logic framework. On the one hand, it is shown that models of computations including concurrent ones such as ECATNets can be naturally and directly expressed as rewrite theories in rewriting logic [2]. On the other hand, it is shown in [14] that linear logic can be subsumed within rewriting logic; in particular a conservative map of entailment systems between linear logic and rewriting logic is given and leads to a very natural correspondence between models of these logics. Thus, it is interesting to achieve, in a rigorous way, the axiomatization of our extended ECATNet using the associated rewrite theory enriched with new operations analogous to some linear logic connectors.
The abstract model of the extended ECATNet is given by the category $\mathcal{C}_{\mathcal{EN}}$ which contains all the objects and morphisms of the category $\mathcal{C}_{\mathcal{N}}$ [2], model of a simple

ECATNet, as well as additional new objects and morphisms provided by the extra structure. The definition of this extra structure is inspired from the categorical interpretation of some linear logic connectors. $\mathcal{C}_{\mathcal{EN}}$ represents all the computations of a system specified with our extended ECATNet as well as more general "idealized" computations. So, in $\mathcal{C}_{\mathcal{EN}}$ we can deduce more complex properties including positive and negative ones.

1.1 Related Works

Linear logic introduced by Girard in [9] has been described as a "resource conscious" logic by Marti-Oliet and Meseguer [14]; in its proofs occurrences of propositions cannot be used more than once or disappear unless they are explicitly created or used up by the inference rules. Researchers were not long in spotting a relationship with Petri nets where there are similar ideas. Places in a Petri net hold to certain non-negative multiplicities forming a multiset of places, traditionally called a marking; as transitions occur, multiplicities of places are consumed and produced in accord with a dynamic behaviour of nets, formalised in the so-called "token game".

In [11], [6], [1], and [10], it is shown that places are like atomic propositions in linear logic and transitions like proof rules. Essentially, it is shown that an atomic formula A can be thought of as a token in a place named A, and a tensor formula $A_1 \otimes \ldots \otimes A_n$ as a marking, i.e., a distribution of tokens on the places $A_1, \ldots, A_n$. The proof of a sequent of the form $A_1, \ldots, A_n \vdash B$ is then a computation from the marking $A_1 \otimes \ldots \otimes A_n$ to B. A specific net is described by a set of external axioms (*a tensor theory*) and the dynamic behaviour of the net is described by the inference rules of the tensor fragment.

Alongside the work on Petri nets and linear logic, came the realisation that models of linear logic had arisen in the form of *quantales* [9]. Indeed, Winskel and Enberg in [7] point out a straightforward way in which a Petri net induces a quantale and so becomes a model for linear logic. Their paper provides evidence that linear logic with the right notion of satisfaction, can be a reasonably expressive specification logic for parallel processes. More explicitely, in [7] Engberg and Winskel associate to a given Petri net N a quantale $Q[N]$ whose elements are the set of markings downwards closed with respect to the reachability relation. Each atomic proposition A is denoted by a set of reachable markings of the corresponding place A. General formulas of linear logic will then be statements about the reachability of markings. The work of [8] contrasts with the above approach to linear logic and Petri nets in which, only small fragments of the logic such as tensor fragment [11] are applied; thus the logic becomes rather inexpressive, and in particular cannot capture negative properties.

In [14], authors extend the quantale models of linear logic to the categorical ones, in the sense that quantales are a special case of linear categories whose category structure is a poset. They proceed by letting Petri nets freely generate a linear category and then interpreting linear logic in that setting. Consequently, a Petri net can be reinterpreted as a linear logic theory which has a linear category as its associated initial model. The reachable states of the Petri net become propositions in linear logic, and are interpreted as objects in this linear category; the computations of the Petri net become deductions in linear logic and are interpreted as morphisms in the categorical semantics.

The purpose of this work is to extend the correspondence between linear logic and Petri nets to a correspondence between linear logic and high-level nets with special emphasis on ECATNets. The main motivation of this correspondence is the need to have an ECATNet model which is more expressive and can capture complex properties including positive and negative ones.
A relationship between algebraic high-level nets of Reisig [16] and intuitionistic predicate linear logic has been already pointed out in [13]. Lilius establishes in [13] a correspondence between the two models in several steps. First he shows how a Petri net gives rise to a model of linear logic [7] and proves that the construction is functorial. Then he shows how an algebraic high-level net gives rise to a Petri net [16]. This construction is also proved to be functorial. Finally, the result desired is the composition of two functors.

A similar approach can be directly used since an ECATNet is naturally transformed in a simple Petri net model [3]. However, regarding the natural definition of an ECATNet in rewriting logic, and exploiting the similarity between the categorical models of linear logic and those of ECATNets which are also categories in the rewriting logic framework, it becomes much simpler to achieve, in a rigorous way, the axiomatization of the correspondence between the two models.

1.2 Paper outline

The paper is structured as follows. We start with a review of linear logic and its model theory. Then we present ECATNets using rewriting logic as a unified framework. Afterwards, an extended ECATNet modelis introduced with its semantics. The resulting model construction is based on the categorical interpretation of some linear logic connectives in the rewriting logic framework.

2 Linear Logic

Linear logic [9] is essentially a Gentzen Calculus of Sequents without *weakening* and *contraction* rules, and with a duplication of the usual logical connectives of conjunction and disjunction, naturally suggested by the lack of such structured rules. This means that neither useless premises can be freely added during the inference, nor different occurrences of the same formula in the premises can be identified: each hypothesis is "used" once and only once. In this sense, logical formulae loose their abstract, platonistic contenance of truth values or types, gaining the more concrete nature of "resources" or "states". Moreover, any step of logical deduction modifies the state of its premises. This is very appealing for computer science applications, and in particular for concurrent computations since it puts the emphasis on dynamics.

In linear logic, two conjunctions $\otimes$ (*times*) and & (*with*) coexist. Both conjunctions express the availability of two actions; but in the case of $\otimes$, both will be done, whereas in the case of &, only one of them will be performed (but we shall decide which one). There are also two disjunctions in linear logic: $\oplus$ (*plus*), dual of &, expresses the choice of one action between two possible types of actions, and ∂ (*par*,)dual of $\otimes$, expresses a dependency between two types of actions.

The most important linear connective is linear negation $(\_)^{\perp}$. It behaves like transposition in linear algebra ($A \multimap B$ will be the same as $B^{\perp} \multimap A^{\perp}$), i.e., linear negation expresses a duality.

The connectives $\otimes$, ∂, $\multimap$, together with the neutral elements I (w.r.t. $\otimes$) and $\perp$ (w.r.t. ∂) are called *multiplicatives.*
The connectives & and $\oplus$, together with the neutral elements $\top$ (w.r.t. &) and 0 (w.r.t. $\oplus$) are called *additives.*

The absence of the rules for *weakening* and *contraction* is compensated, to some extent, by the addition of the logical operators *of-course* (!) and *why-not* (?) which are presented in [9]. We are interested in presenting a *propositional linear logic,* without modalities $(!, ?)$, but including *negation.*

A linear formula is generated by the binary connectives $\otimes, \oplus, \multimap, \partial, \&$ and by the unary operation $(\_)^\perp$ from a collection of propositional constants, including the logical constants $I, \perp, \top, 0$.

A sequent in linear logic has the syntactic structure $\Gamma \vdash \Delta$ where Γ and Δ are finite (possibly empty) lists of linear formulas. The inference rules of the calculus formalize the process of construction of complex proofs by means of simpler ones.

A linear theory T is given by a collection of propositional constants S and a collection of S-sequents (linear sequents whose formulas are constructed from the constants in S and the logical constants) called axioms.

Given a linear theory $T = (S, Ax)$, an S-sequent $\Gamma \vdash \Delta$ belongs to the closure of T, denoted T^*, if it can be derived from the axioms Ax using the following inference rules:

Axiom: $$\frac{}{A \vdash A}(Id)$$

Cut rule: $$\frac{\Gamma \vdash A \quad A, \Gamma' \vdash B}{\Gamma, \Gamma' \vdash B}(cut)$$

Structural rules:

$$\frac{\Gamma, A, B, \Gamma' \vdash \Delta}{\Gamma, B, A, \Gamma' \vdash \Delta}(exchange,\text{left}) \qquad \frac{\Gamma \vdash \Delta, A, B, \Delta'}{\Gamma \vdash \Delta, B, A, \Delta'}(exchange,\text{right})$$

Logical rules:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'}(\otimes,\text{right}) \qquad \frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}(\otimes,\text{left})$$

$$\frac{}{\vdash I}(I,\text{right}) \qquad \frac{\Gamma \vdash \Delta}{\Gamma, I \vdash \Delta}(I,\text{left})$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \& B, \Delta}(\&,\text{right}) \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma, A \& B \vdash \Delta}(\&,\text{left})$$

$$\frac{}{\Gamma \vdash \top, \Delta}(\top,\text{right})$$

$$\frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \oplus B, \Delta}(\oplus,\text{right}) \qquad \frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \oplus B \vdash \Delta}(\oplus\ ,\text{left})$$

$$\frac{}{\Gamma, 0 \vdash \Delta}(0,\text{left})$$

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta}(\multimap,\text{right}) \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \multimap B \vdash \Delta, \Delta'}(\multimap,\text{left})$$

$$\frac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A\partial B, \Delta}(\partial,\text{right}) \qquad \frac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A\partial B \vdash \Delta, \Delta'}(\partial,\text{left})$$

$$\frac{}{\Gamma \vdash}(\bot,\text{left}) \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \bot, \Delta}(\bot,\text{right})$$

As an example of a proof, we can derive the rule

$$\frac{\Gamma \vdash A \multimap B}{\Gamma, A \vdash B}$$

from the logical rule ($\multimap$,left) and the cut one by:

$$\frac{\Gamma \vdash A \multimap B \quad \dfrac{\overline{A \vdash A} \quad \overline{B \vdash B}}{A, A \multimap B \vdash B}(\multimap, \text{left})}{\Gamma, A \vdash B}(cut)$$

2.1 Categorical Interpretation of Linear logic

Having already introduced the syntax and the proof theory of linear logic, we are now ready to present its linear categorical semantics which has been mostly developed by Lafont [12] and Seely[17], with contributions of several other people ([15], [14], [1]). The linear logic model which is adopted in this paper is a simple axiomatization of linear category known as a closed symmetrical monoidal category with the notion of a dualizing object. We recall here the notion of a closed symmetrical monoidal category and dualizing object without giving all the details that can be found in [14].

Definition 2.1 *(Symmetrical Monoidal Category) A symmetrical monoidal category is a category $\mathcal{C}$ with a functor $\_\otimes\_ : \mathcal{C} \times \mathcal{C} \longrightarrow \mathcal{C}$ (called tensor product), and an object I such that:*

(ass): $A \otimes (B \otimes C) \cong (A \otimes B) \otimes C$
(com): $A \otimes B \cong B \otimes A$
(e): $I \otimes A \cong A$

are natural isomorphisms, called structural isomorphisms.

More generally, any category with finite products or coproducts is a symmetrical monoidal category. As we shall see later, the linear logic connective $\otimes$ is interpreted in the models as a monoidal product of this kind.

Definition 2.2 *A closed symmetrical monoidal category is a symmetrical monoidal category $(\mathcal{C}, \otimes, I, ass, com, e)$ such that for each object A of $\mathcal{C}$, the functor $\_ \otimes A : \mathcal{C} \longrightarrow \mathcal{C}$ has a right adjoint $A \multimap \_ : \mathcal{C} \longrightarrow \mathcal{C}$, that is, for all objects A, B, C in $\mathcal{C}$, we have a natural isomorphism $\varphi : Hom_{\mathcal{C}}(B \otimes A, C) \longrightarrow Hom_{\mathcal{C}}(B, A \multimap C)$*

The intuitive interpretation of $A \multimap B$ is the internalization of the morphisms from A to B as an object of $\mathcal{C}$. We have chosen the notation $A \multimap B$ to suggest that linear implication will be interpreted in the models by the functor $\_ \multimap \_$.

Definition 2.3 *Given a closed symmetrical monoidal category $(\mathcal{C}, \otimes, I, ass, com, e, \multimap)$, an object $\perp$ in $\mathcal{C}$ is a dualizing object if, for every object A in $\mathcal{C}$, the natural morphism:*

$$d_{A,\perp} : A \longrightarrow (A \multimap \perp) \multimap \perp$$

is an isomorphism.

The choice of notation is justified by the intimate connection between dualization and linear logic negation. Motivated by the correspondence with linear logic, we are interested in the connective ∂ which is the dual of $\otimes$.

Proposition 2.4 *Let $(\mathcal{C}, \otimes, I, ass, com, e, \multimap, \perp)$ be a category with the dualizing object $\perp$. We define* [1]*:*

1. $A \partial B = (A^{\perp} \otimes B^{\perp})^{\perp}$ *for objects* A, B
2. $f \partial g = (f^{\perp} \otimes g^{\perp})^{\perp}$ *for morphisms* f, g
3. $ass'_{A,B,C} = (id_{A^{\perp}} \otimes d_{B^{\perp} \otimes C^{\perp}})^{\perp}; (ass^{-1}_{A^{\perp},B^{\perp},C^{\perp}}); (\, d^{-1}_{A^{\perp} \otimes B^{\perp}} \otimes id_{C^{\perp}})^{\perp}$
4. $com'_{A,B} = (com_{B^{\perp},A^{\perp}})^{\perp}$
5. $e'_A = (e^{\uparrow}_{\perp} \otimes id_{A^{\perp}})^{\perp}; (e^{-1}_{A^{\perp}})^{\perp}; d^{-1}_A$

Then $(\mathcal{C}, \partial, \perp, ass', com', e')$ is a symmetrical monoidal category.

Again, our notation is chosen to suggest that the "par" connective ∂ will be interpreted in the models by the functor $\_\partial\_$.

Our choice of notation throughout this paper is motivated by our desire to emphasize the connections with linear logic. Indeed, a natural categorical semantics for linear logic will interpret conjunction $\otimes$ and linear implication $\multimap$ as tensor product and internal Hom, respectively, in a closed symmetric monoidal category. Classical negation $(\_)^{\perp}$ is then interpreted as dualization with a dualizing object $\perp$. Similarly, the additives ∂ and $\oplus$ are interpreted as products and coproducts respectively. This motivates the definition of a linear category, as the natural notion of a model for classical linear logic, proposed by Marti-Oliet and Meseguer in [14]. Several examples of linear categories are also discussed in [14].

Definition 2.5 *A linear category $(\mathcal{C}, \otimes, I, ass, com, e, \multimap, \perp, \&, \top)$ is a category with dualizing object $(\mathcal{C}, \otimes, I, ass, com, e, \multimap, \perp)$ and chosen finite products, i.e., a final object $\top$ and for any objects A, B, a binary product denoted $A \& B$.*

Given a linear category $\mathcal{C}$, a theory $T = (S, Ax)$, and an assignment of an object $\mid s \mid$ in $\mathcal{C}$ to each constant $s \in S$, it is clear that we can interpret any S-formula A as an object $\mid A \mid$ in $\mathcal{C}$. Then we will associate with each derivation of an S-sequent $A_1, \ldots, A_n \vdash B_1, \ldots, B_m$ a corresponding morphism $\mid A_1 \mid \otimes \cdots \otimes \mid A_n \mid \longrightarrow \mid B_1 \mid \partial \cdots \partial \mid B_m \mid$ in $\mathcal{C}$.

[1] If $f : B \otimes A \longrightarrow C$ is a morphism in $\mathcal{C}$, we write $f^{\uparrow}$ for $\varphi(f) : B \longrightarrow A \multimap C$, called the *currying* of f.

Definition 2.6 *Given a linear theory $T = (S, Ax)$, there is a linear category $\mathcal{L}[T]$ whose objects are the S-formulas, and whose morphisms are equivalence classes of derivations of S-sequents $\Gamma \vdash \Delta \in T^*$ with respect to the congruence generated by the collection of equations that a category needs to satisfy in order to be a linear category.*

An important breakthrough has occurred with the recent introduction of linear logic by Girard. This new logic has from its beginning been recognized as well suited for computer science applications. Girard and his group have initiated some of these applications in the area of operational semantics and logic programming, and there are at present interesting others new developments [14]. Much effort has been devoted by several researchers to the study of the relationships between linear logic and concurrency through categories. Our own research is situated in this area, and the following sections develop an extended ECATNet specification for concurrent systems taking profit of the categorical semantics of both linear logic and ECATNets.

3 Rewriting Logic framework

In this section, we first recall the definition and the model theory of rewriting logic. Details may be found in [15]. Then, we describe, through an example, how an ECATNet can be regarded as a rewrite theory. Finally, we present some ECATNet properties involving the categorical model of ECATNets.

3.1 Rewriting Logic Review

A *signature* in rewriting logic is a pair (Σ, E) with Σ a ranked alphabet of function symbols and E a set of Σ-equations. Rewriting will operate on equivalence classes of terms modulo the set of equations E.

Given a signature (Σ, E), *sentences* of the logic are sequents of the form $[t]_E \longrightarrow [t']_E$, where t and t' are Σ-terms possibly involving some variables.
A *theory* in this logic, called rewrite theory, is a slight generalization of the usual notion of theory in [15]; in addition, the axioms are allowed to be labelled.

Definition 3.1 *A labelled rewrite theory $\mathcal{R}$ is a 4-tuple $\mathcal{R} = (\Sigma, E, L, R)$, where Σ is a ranked alphabet of function symbols, E is a set of Σ-equations, L is a set called the set of labels, and $R \subseteq L \times T_{\Sigma,E}(X)^2$ is a set of pairs whose first component is a label and whose second component is a pair of E-equivalence classes of terms, with $X = \{x_1,...,x_n\}$ a countably infinite set of variables. Elements of R are called rewrite rules. For a rewrite rule (r, [t], [t']) we use the notation r: [t] $\rightarrow$ [t'].*

Given a labelled rewrite theory $\mathcal{R}$, we say that $\mathcal{R}$ entails a sequent [t] $\rightarrow$ [t'] and write $\mathcal{R} \vdash$ [t] $\rightarrow$ [t'] iff [t]$\rightarrow$ [t'] can be obtained by finite applications of the following deduction rules:

1. *Reflexivity.*
$$\forall [t] \in T_{\Sigma,E}(X), \frac{}{[t] \rightarrow [t]}$$
2. *Congruence.*
$$\forall f \in \Sigma_n, n \in \mathbb{N} : \frac{[t] \rightarrow [t_1']...[t_n] \rightarrow [t_n']}{[f(t_1,...,t_n)] \longrightarrow [f(t_1',...,t_n')]}$$

3. *Replacement.* For each rewrite rule r: [t(x1,...,xn)] → [t'(x1,...,xn)] in R,

$$\frac{[w_1] \ \rightarrow \ [w_1'] \ ... \ [w_n] \ \rightarrow \ [w_n']}{[t(\overline{w}/\overline{x})] \longrightarrow [t'(\overline{w}'/\overline{x})]}$$

where $\overline{w}/\overline{x}$ is a substitution of w_i for $x_i, 1 \leq i \leq n$.

4. *Transitivity.*

$$\frac{[t1] \longrightarrow [t2][t2] \longrightarrow [t3]}{[t1] \longrightarrow [t3]}$$

Rewriting logic is a logic for reasoning correctly about concurrent systems having states, and evolving by means of transitions. Given a labelled rewrite theory $\mathcal{R} = (\Sigma, E, L, R)$, the model for it is a category $\mathcal{T}_{\mathcal{R}}(X)$, whose objects (states) are equivalence classes of terms $[t] \in T_{\Sigma,E}(X)$ and whose morphisms (transitions) are equivalence classes of proof terms representing proofs in rewriting deduction, i.e., concurrent $\mathcal{R}$-rewrites. The category $\mathcal{T}_{\mathcal{R}}(X)$ is one among many other categories that can be assigned to the rewrite theory $\mathcal{R}$. The general notion of model, called an $\mathcal{R}$-system, is also a category $\mathcal{R}$-Sys which is detailed in [15].

A variety of models of concurrency have been obtained as special cases of concurrent rewriting. A natural way of studying specializations of this kind is to impose restrictions on the rewrite theories being used. A wide variety of models of computations and languages such as Petri nets, actors, λ-calculus, CCS, event structures, can be naturally expressed in rewriting logic. In particular, the application of rewriting logic to algebraic Petri nets has been obtained as a special case of concurrent rewriting. The following section explains the correspondence between the ECATNet concurrent model and rewriting logic.

3.2 ECATNets as rewrite theories

ECATNets have a straightforward and natural expression as labelled rewrite theories. Their distributed states correspond to *markings*, that is, to finite multisets of *structured data,* which are tuples of the form <place, token>. Tokens are axiomatized by algebraic data types. Algebraic markings are also axiomatized by an *associative* and *commutative* multiset union operation ($\otimes$) with *identity* the empty multiset (I). A transition t in an ECATNet is simply a labelled rewrite rule $t : M \longrightarrow M'$ between two multiset markings [2].

Therefore, we can view an ECATNet $\mathcal{N}$ as a rewrite theory $\mathcal{N}$ with the above algebraic axiomatization for its markings and with one rewrite rule per transition, so that firing of a transition exactly corresponds to a rewriting modulo associativity, commutativity and identity with the corresponding rewrite rule.

Clearly, an ECATNet labelled rewrite theory is a static description of what an ECATNet can do. The deduction rules allow us to reason correctly about which general concurrent transitions are possible in a system satisfying such a description. The meaning of the theory should be given by a categorical model, noted $\mathcal{C}_{\mathcal{N}}$, of its actual behaviour. This $\mathcal{C}_{\mathcal{N}}$ construction is a particular case of the general construction of $\mathcal{T}_{\mathcal{R}}$ in [15].Then, in this model, objects are represented by states (terms of sort *markings*), and the finite concurrent computations are described by arrows [2].

The following example is intended to illustrate the use of simplified ECATNets. The reader interested in more details may consult [5], [4].

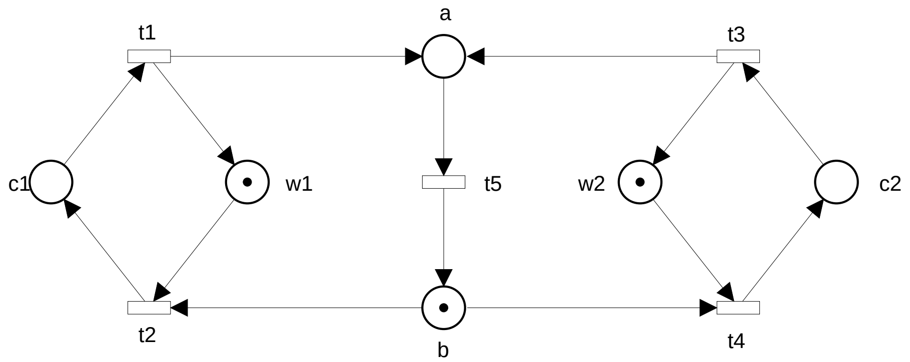

FIG. 1. An ECATNet specifying the Mutual Exclusion Problem

Example 3.2 *(Mutual exclusion)*

Consider the net $\mathcal{N}$ of figure 1. where the marking $<w1, \cdot>$ indicates that the first process, P1, is working outside its critical section, c1, and similarly for the other process, P2. The resource corresponding to b is used to ensure mutual exclusion of the critical sections $(c1, c2)$ after a process has been in its critical region it returns a resource, a, which then be transformed into b for the next turn. The initial marking, M_0, will be $M_0 =<b, \cdot> \otimes <w1, \cdot> \otimes <w2, \cdot>$

For simplicity, tokens considered in our simplified ECATNet are "indistinguishable" black dots instead of algebraic terms.

Following the previous ideas that go back to the work of associating an ECATNet to a rewrite theory [2], we present an alternative description of the above ECATNet example. All the specifications are given in a Maude-like notation [15].

Mod MEX is
extending $TOKENS$.
Subsorts:
place, marking $<$ markings
ops:
$c1$, $c2$, $w1$, $w2$, a, b :$\longrightarrow$ place
I:$\longrightarrow$marking
$<$ _, _ $>$: place,tokens$\longrightarrow$marking
_ $\otimes$ _ : markings,markings$\longrightarrow$markings [ACI:I]
rl1/$t1$:$<c1,\cdot>\longrightarrow <w1,\cdot>\otimes<a,\cdot>$
rl2/$t2$:$<b,\cdot>\otimes<w1,\cdot>\longrightarrow<c1,\cdot>$
rl3/$t3$:$<c2,\cdot>\longrightarrow <w2,\cdot>\otimes<a,\cdot>$
rl4/$t4$:$<b,\cdot>\otimes<w2,\cdot>\longrightarrow<c2,\cdot>$
rl5/$t5$:$<a,\cdot>\longrightarrow<b,\cdot>$
endm

We use the existing theory $TOKENS$ which specifies and generates the elements of sort tokens representing arc, place and transition inscriptions (black dots in this *case).*

A concurrent computation in this system is described by the possible rewriting logic deduction with rules: rl1, rl2,..., rl5. The fact that $\mathcal{N}$ *evolves from marking*

$M_0 = <b, \cdot> \otimes <w1, \cdot> \otimes <w2, \cdot>$

to marking

$M = <c1, \cdot> \otimes <w2, \cdot>$

corresponds directly to the following MEX*-rewrite (proof in rewriting logic):*

$MEX \vdash <b, \cdot> \otimes <w1, \cdot> \otimes <w2, \cdot> \longrightarrow <c1, \cdot> \otimes <w2, \cdot>$

using rl2.

3.3 Expressing Properties of ECATNets

The precise semantics of ECATNets follow from their integration in the unified framework of rewriting logic. All possible behaviours of an ECATNet, seen as a rewrite theory $\mathcal{N}$, can be represented by the categorical abstract model $\mathcal{C}_{\mathcal{N}}$ (a particular case of $\mathcal{T}_{\mathcal{R}}$ [15]) which gives a semantics to the ECATNet concurrent computations.

Given the meaningful semantics interpretation of the ECATNet behaviour [2]; it is shown that proofs in rewriting logic and ECATNet computations are formally identical. The potential benefit of this correspondence is that some properties of concurrent systems specified with ECATNets, can be checked and deduced in a natural way.

Example 3.3 *(some properties of* MEX*-rewrite theory)*

A provable sequent $A \longrightarrow B$ *in* MEX*-rewrite theory corresponds exactly to the path* $A \Longrightarrow^* B$ *in the reachability graph of the net in figure 1. Typically, the following simple properties:*

- *a state* S *is reachable from the initial state* S_0*,*
- *from the initial marking it is possible to reach a marking where place a is marked,*
- *once a is marked it is possible to reach a marking where place b is marked,*

can be naturally expressed by investigating if there exists the following provable sequents in rewriting logic:

- $MEX \vdash S_0 \longrightarrow S$
- $MEX \vdash S_0 \longrightarrow <a, \cdot> \otimes M$
- $MEX \vdash <a, \cdot> \otimes M \longrightarrow <b, \cdot> \otimes M'$

In particular, we can express that:

- *P1 can enter its critical section from the initial marking* $M_0 = <b, \cdot> \otimes <w1, \cdot> \otimes <w2, \cdot>$ *by proving:* $MEX \vdash <w1, \cdot> \longrightarrow <c1, \cdot>$.
- *when P1 is in critical section and P2 is working, it is possible that P2 can later come into its critical section with P1 working by proving:* $MEX \vdash <c1, \cdot> \otimes <w2, \cdot> \longrightarrow <c2, \cdot> \otimes <w1, \cdot>$.

In the next section, we shall see how to express more complex properties including "negative" ones, e.g.:

- if the system is in a "working state" then both processes may enter their critical section.
- both processes cannot be in their critical sections at the same time.

4 Extending ECATNet Rewrite Theory

The objective of this section is to extend the ECATNet model with the introduction of new objects and morphisms that could be useful for specification and reasoning purposes. The general idea is to enrich the ECATNet rewrite theory with the definition of some operations whose interpretation is similar to that of linear logical connectors: $\&, \multimap, \oplus, \partial$. The naturalness and directness with which these operations can be syntactically specified is due to the great flexibility and generality of rewriting logic. The abstract syntax of the new ECATNet states can be represented as another algebraic data type of the same sort *markings* but does not have "real" counterparts as states of the net. Therefore, we can think of these states as "idealized" states that could be useful for reasoning purposes. For example, we propose to interpret a state of the form $M1 \multimap M2$ as a kind of conditional state or non-concluded state as in [1].

Example 4.1 *(Extended ECATNet)*
The extended ECATNet rewrite theory specifying the previous example (figure 1) is given below. It illustrates the syntax of the additional operations.

fmod $EXT\text{-}MEX$ *is*
  extending MEX.
  ops:
   $0, \top, \bot \longrightarrow$ *markings*
   $\_ \multimap \_$ *: markings,markings* $\longrightarrow$ *markings*
   $\_\&\_$ *: markings,markings* $\longrightarrow$ *markings [ACI:* $\top$*]*
   $\_\partial\_$ *: markings,markings* $\longrightarrow$ *markings [ACI:* $\bot$*]*
   $\_ \oplus \_$ *: markings,markings* $\longrightarrow$ *markings [ACI:* 0*]*
   $(\_)^{\bot}$ *: markings* $\longrightarrow$ *markings*
   vars A, B*: markings*
   eq $(A \otimes B)^{\bot} = A^{\bot} \partial B^{\bot}$
   eq $(A \oplus B)^{\bot} = A^{\bot} \& B^{\bot}$
   eq $(A \partial B)^{\bot} = A^{\bot} \otimes B^{\bot}$
   eq $(A \& B)^{\bot} = A^{\bot} \oplus B^{\bot}$
   eq $A \multimap B = (A \otimes B^{\bot})^{\bot}$
   eq $(A \multimap B)^{\bot} = B^{\bot} \multimap A^{\bot}$
   eq $I^{\bot} = \bot$
   eq $\bot^{\bot} = I$
   eq $\top^{\bot} = 0$
   eq $0^{\bot} = \top$
endfm

In order to complete the construction of the enriched ECATNet, we need a way of getting a categorical model of it. The style of our formulation exploits the categorical interpretation of linear logic connectives, and the similarity between models of linear logic and those of ECATNets which are also categories in rewriting logic framework

Since each rewrite theory has an initial model [15] that mathematically characterizes the concurrent system as a category whose objects are the system states and whose morphisms are the system concurrent computations, the abstract model of our extended ECATNet rewrite theory $\mathcal{C}_{\mathcal{EN}}$ is an extension of the category $\mathcal{C}_{\mathcal{N}}$ (model of the corresponding simple ECATNet rewrite theory) with a closed structure (definitions 2.2 and 2.3, proposition 2.4). It contains all the objects and morphisms of $\mathcal{C}_{\mathcal{N}}$ – corresponding to markings and computations of the ECATNet – as well as additional new objects and morphisms provided by the extra structure. In the following we will first give the categorical interpretation of these new objects (states) and morphisms (computations) and then explain their computational interpretation.

4.1 Categorical Interpretation of $\mathcal{C}_{\mathcal{EN}}$

¿From a categorical point of view, $\mathcal{C}_{\mathcal{EN}}$ is a linear category ($\mathcal{C}$, $\otimes$, I, ass, com, e, $\multimap$, $\perp$, $\&$, $\top$) such that the category $\mathcal{C}$ with the functor $\_ \otimes \_ : \mathcal{C} \times \mathcal{C} \longrightarrow \mathcal{C}$ and the object I verifying the structural isomorphisms (ass, com, e) correspond exactly to the category $\mathcal{C}_{\mathcal{N}}$[2]. In particular, the tensor product $\_ \otimes \_$ in $\mathcal{C}$ corresponds to the functor $\_ \otimes \_$ which has already been defined in $\mathcal{C}_{\mathcal{N}}$. But the categorical interpretation of $\multimap$, $\perp, \&$,and $\top$ must be given.

The interpretation of $\multimap$ as an internal Hom (right adjoint to $\_ \otimes \_$) suggests the addition of the following rules:

1. For $[a], [b]$,and $[c]$, terms of sort *markings*,

$$\frac{\alpha : [c] \otimes [a] \rightarrow [b]}{\alpha^{\uparrow} : [c] \longrightarrow [a] \multimap [b]}$$

2. For $[a], [b]$, $[c]$, $[d]$,and $[e]$, terms of sort *markings*,

$$\frac{\alpha : [d] \rightarrow [a] \quad \beta : [b] \otimes [e] \rightarrow [c]}{(\alpha, \beta)^{\uparrow} : [d] \otimes [e] \otimes [a] \multimap [b] \longrightarrow [c]}$$

We need to add a special object $\perp$ (false), the dualizing one, such that the negation $(\_)^{\perp}$ is equivalent to $\_ \multimap \perp$, so for any object $[a]$ in $\mathcal{C}_{\mathcal{EN}}$ (or simply $\mathcal{C}_{\mathcal{N}}$) there is a canonical isomorphism $[a] \cong ([a] \multimap \perp) \multimap \perp$. From a categorical point of view this is interpreted by the addition of the corresponding rule (rule 9 in the following).

Moreover, in such a category the functors $\_\partial\_$ and $\_ \oplus \_$ can be interpreted respectively by:

- $[a]\partial[b] = ([a]^{\perp} \otimes [b]^{\perp})^{\perp}$ $(= [a]^{\perp} \multimap [b])$,
  $\_\partial\_$ is a dual of $\_ \otimes \_$. So, $\perp$ is the unit for $\_\partial\_$ in the same way that I is the unit for $\_ \otimes \_$.
- $[a] \oplus [b] = ([a]^{\perp} \& [b]^{\perp})^{\perp}$.
  $\_ \oplus \_$ is a dual of $\_\&\_$. So, 0 is the unit for $\_ \oplus \_$ in the same way that $\top$ is the unit for $\_\&\_$.

Consequently, the following rules are added to $\mathcal{C}_{\mathcal{EN}}$ in order to generate additional proofs.

3. For $[a], [b]$ and $[c]$, terms of sort *markings*,

$$\frac{\alpha : [a] \longrightarrow [c] \ \beta : [b] \longrightarrow [c]}{(\alpha \oplus \beta, l) : [a] \oplus [b] \longrightarrow [c]}$$

4. For $[a]$ and $[b]$, terms of sort *markings*,

$$\frac{}{([a]^{\oplus}, right) : [a] \longrightarrow [a] \oplus [b]}$$

5. For each $[c]$, term of sort *markings*,

$$\frac{}{([c]^{0}, right) : 0 \longrightarrow [c]}$$

6. For $[a], [b]$ and $[c]$, terms of sort *markings*,

$$\frac{\alpha : [a] \longrightarrow [b] \ \beta : [a] \longrightarrow [c]}{(\alpha \& \beta, right) : [a] \longrightarrow [b] \& [c]}$$

7. For $[a]$ and $[b]$, terms of sort *markings*,

$$\frac{}{([a]^{\&}, left) : [a] \& [b] \longrightarrow [a]}$$

8. For each $[a]$, term of sort *markings*,

$$\frac{}{([a]^{\top}, left) : [a] \longrightarrow \top}$$

9. For each $[a]$, term of sort *markings*,

$$\frac{}{([a]^{\perp}, left) : [a] \otimes [a]^{\perp} \longrightarrow \perp}$$

10. For each $[a]$, term of sort *markings*,

$$\frac{}{([a]^{\perp}, right) : I \longrightarrow [a] \partial [a]^{\perp}}$$

We will note the similarity between the above proof generating rules and logical rules of linear connectives. The previous $\mathcal{C}_{\mathcal{EN}}$ presentation ensures that the model of the extended ECATNet rewrite theory is an $\mathcal{EN}$-system which is a particular case of the general notion of rewrite theory model called an $\mathcal{R}$-system (see [14]).

4.2 Computational Interpretation of $\mathcal{C}_{\mathcal{EN}}$

The category $\mathcal{C}_{\mathcal{EN}}$ represents all the computations of the system, specified with an extended ECATNet, as well as more general "idealized" computations.

In particular, we interpret a state of the form $M_1 \multimap M_2$ as a conditional state. Consider again the ECATNet of figure 1, the state:

$<b, \cdot> \otimes <w_1, \cdot> \longrightarrow <c_1, \cdot>$

means that when $P1$ is working and resource b is available, the process can enter its critical section. Following the usual definition of a transition firing, nothing can happen from the state $<b, \cdot>$. However, using rule 1, we can derive the sequent:

$$<b,\cdot>\longrightarrow<w_1,\cdot>\multimap<c_1,\cdot>.$$

The firing of this computation results in a conditional state $<w_1,\cdot>\multimap<c_1,\cdot>$. The idea is that although some states have been rewritten, some other states ($<w_1,\cdot>$ in this particular example) are needed in order to fire transition $t2$ (rl2 in MEX), by means of the computation:

$$(<w_1,\cdot>\multimap<c_1,\cdot>)\otimes<w_1,\cdot>\longrightarrow<c_1,\cdot>$$

These new non-concluded states, together with the associated computations, allow the observation of computations at a lower level of atomicity where states can be independently rewritten.

We interpret the operations & and $\oplus$ as external and internal choice, respectively. Let us consider again the simple ECATNet in figure 1 with rewriting sequents:

s1: $<w_1,\cdot>\otimes<b,\cdot>\otimes<w_2,\cdot>\longrightarrow<c_1,\cdot>\otimes<w_2,\cdot>$

s2: $<w_1,\cdot>\otimes<b,\cdot>\otimes<w_2,\cdot>\longrightarrow<c_2,\cdot>\otimes<w_1,\cdot>$

Intuitively these sequents mean that being in working state, $P1$ can enter its critical section c_1 and also being in the same state $P2$ can enter its critical section c_2. Using rule 6, we can derive from $s1$ and $s2$ the sequent:

$$<w_1,\cdot>\otimes<b,\cdot>\otimes<w_2,\cdot>\longrightarrow(<c_1,\cdot>\otimes<w_2,\cdot>)\&(<c_2,\cdot>\otimes<w_1,\cdot>)$$

A very appealing interpretation of this sequent is that it indicates the possibility of a *choice* between $P1$ entering its critical section or $P2$ entering its critical section, but not both. Moreover, this choice is *external* in a sense that the user of this specification can make the decision. On the other hand, the rule rl5 in MEX and the rule 4, allow the derivation of the sequent:

$$< a,\cdot >\longrightarrow< b,\cdot >\oplus< w_1,\cdot >$$

This sequent can also be interpreted as, being in state $<a,\cdot>$, we can have the possibility to derive the state $<w_1,\cdot>$ or the state $<b,\cdot>$, but now the choice is *internal*, in the sense that the system "decides" while the user cannot make any decision.

We have noticed that in ECATNets, one useful interpretation of the extra structure $\perp$ is the set of markings (states) which cannot be reached from the empty marking denoted I. Since negation can be expressed in terms of $\perp$ and $\multimap$ by $[a]^{\perp}=[a]\multimap\perp$, we define that a state B is not reachable from a state A by: $A\nrightarrow B$ (or $I\nrightarrow A\multimap B$), i.e., $A\multimap B\longrightarrow\perp$.

The interesting consequence of this particular choice is that whatever property we could state before in terms of provability of a rewrite proof $A\longrightarrow B$ can be stated negatively as $A\nrightarrow B$. Let us consider again the ECATNet in figure 1, we can now verify that the processes, $P1$ and $P2$ cannot get into their critical sections at the same time by proving:

$$M_0\nrightarrow<c1,\cdot>\otimes<c2,\cdot>$$

which is equivalent to the sequent:

$$I\nrightarrow M_0\multimap<c1,\cdot>\otimes<c2,\cdot>$$

in other words:

$M_0 \multimap <c1,\cdot> \otimes <c2,\cdot> \longrightarrow \perp$

This $EXT\text{-}MEX$ rewrite rule (rewriting logic proof) can be deduced as follows. Suppose that we start in the following state:

$M_0 =<b,\cdot> \otimes <w1,\cdot> \otimes <w2,\cdot>$

The first term of this rewrite rule is then equal to:

$<b,\cdot> \otimes <w1,\cdot> \otimes <w2,\cdot> \multimap <c1,\cdot> \otimes <c2,\cdot>$

From this state we can fire concurrently transitions $t2$ and $t3$, and obtain the state:

$<c1,\cdot> \otimes <w2,\cdot> \multimap <c1,\cdot> \otimes <w2,\cdot> \otimes <a,\cdot>$

which is the same as:

$I \multimap <a,\cdot>$

or more precisely:

$<a,\cdot>$

In addition, we know also that:

$I \not\multimap <a,\cdot>$

i.e., $<a,\cdot> \longrightarrow \perp$

Consequently, the corresponding property is verified.

$<b,\cdot> \otimes <w1,\cdot> \otimes <w2,\cdot> \not\multimap <c1,\cdot> \otimes <c2,\cdot>$

5 Conclusion

The main objective of this paper was to explore what could be the basis of a new approach for an ECATNet model-based diagnosis of concurrent systems. The principle of our approach was to use the correspondence between linear logic and ECATNets, a form of algebraic high-level nets. Our formulation has been realized by exploiting the similarity between categorical models of linear logic and those of ECATNets which are also categories in the rewriting logic framework. A fruitful consequence of this correspondence is that ECATNet model can be naturally extended with additional new objects and morphisms in order to have a more expressive model-based diagnosis of concurrent systems. The categorical interpretation of this extra structure was inspired from that of some linear connectives. In particular, the useful interpretation of the extra object $\perp$, provided a richer specification language, based on ECATNets, for the study of "negative" properties.
The obtained ECATNet model is promising, but significant case studies remain necessary to assess and validate the proposed formalism.

References

[1] A. Asperti, G.L. Ferrari, R. Gorrieri, Implicative formulae in the "Proofs as Computations" Analogy, Seventeenth Annual ACM Symposium on Principles of Programming Languages, San Francisco, 1990.

[2] F. Belala, M. Bettaz, ECATNet Behaviours in Rewriting Logic, Research Report, University of Constantine, Algeria, 09/98.

[3] F. Belala, L. Petrucci, Sémantique des ECATNets en termes de CPNets: Application à un exemple de production, In: Proc. MOSIM'97,(Hermès, France, 1997).

[4] M. Bettaz, M. Maouche: Modeling of Object Based Systems With Hidden Sorted ECATNets. In: P. Dowd, E. Gelenbe (eds.): MASCOTS'95, Durham, North-Carolina.

[5] M. Bettaz, G. Reggio: A SMoLCS Based Kit for Defining the Semantics of Algebraic High-Level Nets. In: H. Ehrig, F. Orejas (eds.): Recent Trends in Data Type Specification. Lecture Notes in Computer Science 785, Springer-Verlag, 1994, pp. 98-112.

[6] C. Brown, Relating Petri Nets to Formulae of Linear Logic, Technical Report ECSLFC 89-87, University of Edinburgh, 1989.

[7] U. Egberg, G. Winskel, Petri Nets as Models of Linear Logic, Lecture Notes in Computer Science Vol. 431, CAAP'90 , Copenhagen, Denmark, Springer-Verlag, 1990.

[8] U. Egberg, G. Winskel, Linear Logic on Petri Nets, Lecture Notes in Computer Science Vol. 803, A Decade of oncurrency Reflections and Perspectives, The Netherlands, Springer-Verlag, 1993.

[9] J.-Y. Girard, Linear Logic, Theoretical Computer Science, 50(1987), pp. 1-102.

[10] F. Girault, B Pradin-Chézalviel, L.A. Kunzle, R. Valette, Linear Logic as a tool for reasoning on a Petri Net model, Proceeding of Symposium on Emerging Technologies and Factory Automation, 1995, INRIA/IEEE, Paris (France).

[11] C. Gunter, V. Gehlot, Nets as Tensor Theories, in Proc. 10 th. International Conference on Application and Theory of Petri Nets, Bonn, 1989.

[12] Y. Lafont, T. Streicher, Games Semantics for Linear Logic, in Proc. Sixth Annual IEEE Symposium on Logic in Computer Science, Amsterdam, July 91, pp. 43-50.

[13] J. Lilius, High-level Nets and Linear Logic, Lecture Notes in Computer Science Vol. 616, Application and Theory of Petri Nets, Springer-Verlag, 1992.

[14] N. Marti-Oliet, J. Meseguer, From Petri Nets to Linear Logic, Lecture Notes in Computer Science Vol. 389, Category Theory and Computer Science, Springer-Verlag, 1989.

[15] N. Marti-Oliet, J. Meseguer, Rewriting Logic as a Logical and Semantic Frameworks, Technical Report SRI-CSL, Menlo Park, CA 94025, and Center for the study of Language and Information Stanford University, Stanford, CA 94305.

[16] W. Reisig, Petri Nets and Algebraic Specifications, Theoretical Computer Science, 80(1991), pp. 41-64.

[17] R.A.G. Seely, Linear Logic, *-Autonomous Categories and Cofree Coalgebras, in : J.W. Grayaud A. Seedrov (eds.), Categories in Computer Science and Logic, Boulder, June 87.