Trellis Processes : a Compact Representation for Runs of Concurrent Systems

Eric Fabre

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Trellis Processes : a Compact Representation for Runs of Concurrent Systems

Eric Fabre

N°5554

April 2005

Systèmes symboliques

ISSN 0249-6399 ISRN INRIA/RR--5554--FR+ENG

INRIA
RENNES

Trellis Processes : a Compact Representation for Runs of Concurrent Systems

Eric Fabre*

Systèmes symboliques
Projets Distribcom

Rapport de recherche n ° 5554 — April 2005 — 34 pages

Abstract:

In the true concurrency semantics, runs of a concurrent system are represented as partial orders of events. To represent sets of such runs in a compact manner, traditional approaches rely on branching processes. The maximal branching process of a system, for the prefix relation, is known as the unfolding of that system. Unfoldings (of a finite complete prefix of them) are used for model-checking purposes, reachability analysis, system monitoring or diagnosis, etc. In particular, they enjoy nice factorization properties that allow the development of distributed/modular algorithms, for model checking or diagnosis.

In this communication, we propose an alternate and somehow intermediate representation for runs of concurrent systems, where time is unfolded, but not the conflict relation. In few words, we abandon the requirement that no node of a branching process be in self-conflict, and allow confluence of conflicting histories. We obtain in that way trellis processes, and the time-unfolding of a system. The latter represents the same configuration set as the familiar unfolding, while being more compact in width. Moreover, trellis processes generalize to concurrent systems the usual notion of trellis of an automaton, which forms the support of most on-line monitoring algorithms.

Through a suitable co-reflection of trellis nets in the category of safe nets, we finally prove that time-unfoldings enjoy the same factorization properties as standard unfoldings, which makes these more compact structures possible candidates for distributed verification/diagnosis algorithms.

Key-words: concurrent system, Petri net, true concurrency semantics, branching process, unfolding, event structure, trellis, category theory, product, factorization

(Résumé : tsvp)

This work is supported by RNRT project SWAN, funded by the French Ministry of Research.

* Irisa/Inria, Eric.Fabre@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Processus en treillis : une représentation compacte des trajectoires d'un système concurrent

Résumé : Le dépliage d'un système concurrent représente de façon compacte toutes les exécutions de ce système. Les dépliages sont largement utilisés comme outil de vérification de modèle, et, récemment, comme support pour des applications de diagnostic. Lorsqu'un système concurrent se décompose en produit synchrone de composants plus petits, son dépliage se factorise de la même manière en produit des dépliages de ces composants. Cette propriété est à la base des algorithmes modulaires ou distribués de diagnostic. Nous décrivons ici une autre façon, plus compacte, de représenter l'ensemble des exécutions d'un système concurrent, à l'aide d'une structure que l'on appelle le "treillis". Dans cette structure, le système est déplié le long de l'axe temporel, mais pas dans la dimension des conflits (*i.e.* les choix de transitions dans un système non déterministe). Rappelons que le dépliage déplie ces deux dimensions, ce qui donne des structures branchantes dont la taille augmente rapidement. On montre que les treillis jouissent des mêmes propriétés algébriques que les dépliages, et notamment des mêmes propriétés de factorisation. Cela en fait des candidats très intéressants pour la supervision en ligne (et distribuée) de systèmes concurrents distribués.

Mots-clé : système concurrent, réseau de Petri, sémantique de concurrence vraie, processus de branchement, dépliage, structure d'événements, treillis, théorie des catégories, produit, factorisation

Contents

1 Introduction **4**

2 Nets and unfoldings **5**
2.1 Category of nets 5
2.2 Occurrence nets and unfoldings 7
2.3 Relations between the two categories 9
2.4 Multi-clock nets 10

3 Trellis nets **12**
3.1 Definition 12
3.2 Trellis processes and time unfoldings 15
3.3 Co-reflection of $\overline{Tr}$ into $\overline{Nets}$ 18

4 Relations between nets, trellises and unfoldings **20**
4.1 Co-reflection of $\overline{Occ}$ into $\overline{Tr}$ 20
4.2 Composition of adjunctions 20

5 Application to labeled nets **21**
5.1 Categories of labeled nets 22
5.2 Factorization in elementary trellises 23
5.3 Other properties of trellises 23

6 Variations around the height function **25**
6.1 A causal monotonic function 25
6.2 Different merge rules in the sequential components 26

7 Conclusion **26**

A Some keypoints about adjunctions **29**
A.1 Product 29
A.2 Adjunction 29
A.3 Construction 30
A.4 Properties 31

B On the choice of net morphisms **32**

C Product of labeled trellis nets **32**

D Comparison with "merged processes" **33**

1 Introduction

The unfolding technique was introduced in the early 80's [2, 4, 5] as a convenient way to represent runs of concurrent systems, in the so-called "true concurrency semantics." The idea is to represent trajectories of concurrent systems as partial orders of events, rather than sequences, in order to capture the fact that some events are not causally related and thus could occur in any order, or even at the same time. This semantics has the advantage to reduce drastically the number of relevant runs, since useless interleavings are not taken into account. The unfolding of a system can then be viewed as a compact data structure to store all these possible runs.

Unfoldings were revisited in the 90's as a convenient tool for verification purposes, in particular with the idea of finite complete prefix [7, 9, 11, 12, 14, 15, 16, 18]. More recent work [20] has shown that unfoldings could also be helpful to solve diagnosis problems in concurrent systems, *i.e.* to recover runs explaining observations produced by this system. This approach has been pushed forward to solve the diagnosis problem for *distributed* systems, *i.e.* concurrent systems made of several components with localized interactions. A distributed algorithm was developped for this purpose, based on a factorized representation of the unfolding of the global system [21, 22, 23]. This representation states that the unfolding of a compound system can be expressed as the product of unfoldings of its components, for an appropriate notion of product. This is another powerful idea to compress further the data structure representing all runs of the system. This factorization result can be derived directly, provided one is not reluctant to heavy proofs. But the most simple and elegant derivation is probably due to [4], where it is expressed in the rather abstract (but powerful) framework of category theory.

The objective of this paper is to revisit the notion of unfolding in order to provide another representation for runs of concurrent systems. Specifically, the unfolding performs a *double expansion* of a system. First of all, *time* is unrolled; the unfolding is a partial order of events. But also *conflicts* are expanded: each time there is a choice between n possible events, n branches are created, that will never meet each other again (conflicts are inherited). This justifies the name of *branching process* that is commonly used in the literature. As a consequence of this double expansion, the unfolding generally increases in length and in width. This is striking in the case of an automaton, *i.e.* a sequential machine, for which the unfolding is simply the *tree* of all possible runs, viewed as sequences of events. In communities not concerned by concurrency aspects, another representation of runs has prevailed up to now. Possible runs are represented on a *trellis*, which amounts to unfolding time but *not conflicts*, *i.e.* different (conflicting) runs ending in the same state at time t are merged, since they have the same future[1] (see fig. 1). As a consequence, the data structure representing runs grows along the time axis, but remains bounded in width.

This remark raises several questions. First of all, is it possible to extend this trellis representation to runs of concurrent systems ? Specifically, we would like to capture graphically the concurrency aspects, but abandon the infinite inheritance of conflicts and allow local merges of conflicting histories. The objective is of course to prevent an explosive number of possible histories to keep track of, for example in online monitoring algorithms. Secondly, does a factorization property exist for such structures ? This is a necessary ingredient in view of distributed monitoring/diagnosis procedures. And finally, what are the relations between these *trellis processes* and unfoldings ? Surprisingly, there exist simple answers to these questions, that we explore in this order. We intensively rely on a reasoning proposed in [4], that is almost cut and pasted at several places in this paper, and that we recall in the next section.

Note: A few months after the development of trellis nets, the author came aware of a similar contruction [19], under the name of "Merged Processes." Differences and similarities are analysed in appendix D.

[1] This approach is very popular in the digital communications community, for example, to represent states of a convolutional encoder, and develop the Viterbi algorithm, a maximum likelihood estimation algorithm.

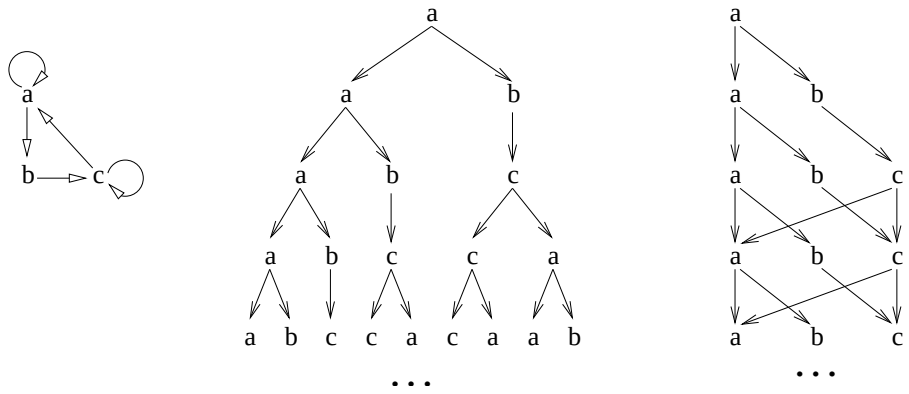

Figure 1: *A sequential machine (left) with* a *as initial state, its unfolding (center), and its trellis (right).*

2 Nets and unfoldings

This section defines our notations for Petri nets (PN), occurrence nets (ON), etc. For more details see [25] or [26]. The contribution of this paper, sections 3 and 4, is based on categorical constructions and results proved by Winskel in [4], that we also need to recall.

2.1 Category of nets

Net. A *net* is a 4-tuple $\mathcal{N} = (P, T, \rightarrow, M^0)$ where P, T are respectively the place and transition sets[2], and $\rightarrow \subseteq (P \times T) \cup (T \times P)$ is the flow relation relating places and transitions[3]. $M^0 : P \rightarrow \{0, 1, 2, \ldots\}$ is a multiset on P representing the initial *marking* of the net, *i.e.* the number of tokens that each place initially holds. Given a node $x \in P \cup T$, we denote by ${}^\bullet x$ and $x^\bullet$ its pre- and post-sets, *i.e.* its immediate predecessors and successors in $\rightarrow$. A transition $t \in T$ is *enabled* (or *activated*) in a marking M, denoted by $M[t\rangle$, iff $\forall p \in {}^\bullet t, M(p) > 0$. Firing t from M yields the new marking $M' = M - {}^\bullet t + t^\bullet$, which we denote by $M[t\rangle M'$. M is a reachable marking iff there exists a sequence $s = (t_1, t_2, \ldots, t_n)$ of transitions (or a *run*) and intermediate markings such that $M^0[t_1\rangle M_1[t_2\rangle M_2 \ldots [t_N\rangle M_N = M$, abbreviated into $M^0[s\rangle M$.

In this paper, we consider only *safe* nets, *i.e.* nets for which places hold at most one token in any reachable marking[4]. We thus identify markings to subsets of P. For technical reasons, specifically the use of recursive procedures in the sequel, we also limit ourselves to safe nets where the number of marked places remains finite, as well as the number of enabled transitions at each marking.

Morphism. To turn the collection of (safe) nets into a category, we need the extra notion of morphism between nets. Let $\mathcal{N}_1, \mathcal{N}_2$ be two nets, with $\mathcal{N}_i = (P_i, T_i, \rightarrow_i, P_i^0)$, $i = 1, 2$. To define a morphism $\phi : \mathcal{N}_1 \rightarrow \mathcal{N}_2$, a natural requirement is that runs of $\mathcal{N}_1$ are turned into runs of $\mathcal{N}_2$. So a first attempt could be as follows:

1. ϕ is a partial function, mapping places to places and transitions to transitions,
2. ϕ preserves the flow relation,
3. the restriction $\phi : P_1^0 \rightarrow P_2^0$ is bijective (on its domain of definition),

[2] We do not require that P and T be finite.

[3] We assume each place is related to one transition at least, and each transition has at least one input place and one output place.

[4] This property can be guaranteed by introducing so-called *complementary places*.

4. $\forall p_1 \in P_1$, if ϕ is defined on p_1, then ϕ is defined on ${}^\bullet p_1$ and on $p_1^\bullet$,

5. $\forall t_1 \in T_1$, if ϕ is defined on t_1, then the restrictions $\phi : {}^\bullet t_1 \to {}^\bullet \phi(t_1)$ and $\phi : t_1^\bullet \to \phi(t_1)^\bullet$ are both bijective (on their domain of definition).

In words, (3) only allows to remove places in the initial marking. Transitions can be removed by ϕ (or even merged) ; but if t_1 is preserved, (5) only allows to erase places in its pre- and post-sets. Finally, places also can be removed, but if a place p_1 is preserved by ϕ, transitions operating on this place can't be removed. This notion is illustrated in fig. 2. It is straightforward to check that if s is a valid run for $\mathcal{N}_1$, then $\phi(s)$ is also a valid run for $\mathcal{N}_2$. Morover, the composition of two morphisms yields another morphism of nets, so we are in good shape to get the desired category of nets.

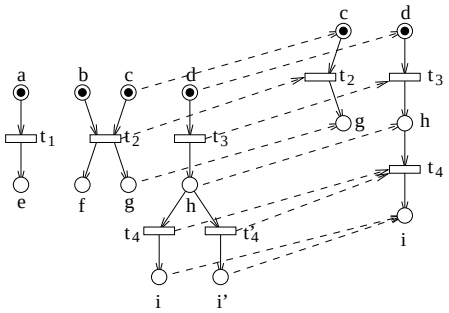

Figure 2: *A typical morphism between two nets (indicated by dashed arrows).*

Unfortunately, this definition is a bit restrictive : the category obtained in that way is not *complete*, and in particular doesn't have a product (see appendix B). The only missing ingredient is the ability of morphisms to *duplicate* places. By reincorporating this possibility, one gets an apparently very different definition since ϕ becomes a *relation* on places, but which is actually a close extension of the previous and intuitive one.

A *morphism* ϕ from $\mathcal{N}_1$ to $\mathcal{N}_2$ is defined as a pair (ϕ_P, ϕ_T) where

1. ϕ_T is a partial function from T_1 to T_2, and ϕ_P a relation between P_1 and P_2,

2. $P_2^0 = \phi_P(P_1^0)$ and $\forall p_2 \in P_2^0$, $\exists! p_1 \in P_1^0 \ : \ p_1 \, \phi_P \, p_2$,

3. if $p_1 \, \phi_P \, p_2$ then the restrictions $\phi_T : {}^\bullet p_1 \to {}^\bullet p_2$ and $\phi_T : p_1^\bullet \to p_2^\bullet$ are total functions,

4. if $t_2 = \phi_T(t_1)$ then the restrictions $\phi_P^{op} : {}^\bullet t_2 \to {}^\bullet t_1$ and $\phi_P^{op} : t_2^\bullet \to t_1^\bullet$ are total functions.

where ϕ_P^{op} denotes the opposite relation to ϕ_P. Notice that points 3 and 4 entail that the pair (ϕ_P, ϕ_T) preserves the flow relation (on its domain of definition). In the sequel, we will simply write ϕ for ϕ_P or ϕ_T, and $\phi(X)$ to denote places in relation with at least one place in X.

The category *Nets* is formed by safe nets provided with the above definition of morphisms.

Product. Let $\mathcal{N}_1, \mathcal{N}_2$ be two nets, their product $\mathcal{N}_1 \times \mathcal{N}_2$ is defined as the triple $(\mathcal{N}, \psi_1, \psi_2)$ where $\mathcal{N} = (P, T, \to, P^0)$ is a net, $\psi_i : \mathcal{N} \to \mathcal{N}_i$ a morphism, and satisfying

1. $P = \{(p_1, \star) : p_1 \in P_1\} \cup \{(\star, p_2) : p_2 \in P_2\}$;
   $\psi_1(p_1, p_2) = p_1$ if $p_1 \neq \star$ and is undefined otherwise, and symmetrically for ψ_2,

2. $P^0 = \psi_1^{-1}(P_1^0) \cup \psi_2^{-1}(P_2^0)$,

3. $T = \{(t_1, \star) : t_1 \in T_1\} \cup \{(t_1, t_2) : t_1 \in T_1, t_2 \in T_2\} \cup \{(\star, t_2) : t_2 \in T_2\}$; $\psi_1(t_1, t_2) = t_1$ if $t_1 \neq \star$ and is undefined otherwise, and symmetrically for ψ_2,

4. $\rightarrow$ is defined by ${}^\bullet t = {}^\bullet\psi_1(t) \cup {}^\bullet\psi_2(t)$ and $t^\bullet = \psi_1(t)^\bullet \cup \psi_2(t)^\bullet$, assuming ${}^\bullet\psi_i(t) = \psi_i(t)^\bullet = \emptyset$ if ψ_i is undefined on t.

In a product, each component preserves its places by the disjoint union in (1), and its transitions by (3), but synchronized transitions are also created, by merging transitions of $\mathcal{N}_1$ and $\mathcal{N}_2$. This definition makes $\times$ the categorical product of *Nets* (see A.1 for a definition, and [6] for a proof).

2.2 Occurrence nets and unfoldings

Occurrence net. The net $\mathcal{O} = (C, E, \rightarrow, C^0)$ is an *occurrence net* (ON) iff it satisfies :

1. $C^0 = \{c \in C \,:\, {}^\bullet c = \emptyset\}$,
2. the *causality* relation $\rightarrow^*$, irreflexive transitive closure of $\rightarrow$, is a partial order, and $\forall x \in C \cup E,\ [x] \triangleq \{x\} \cup \{y \in C \cup E \,:\, y \rightarrow^* x\}$ is finite,
3. $\forall c \in C,\ |{}^\bullet c| \leq 1$,
4. the *conflict* relation $\#$ defined by the two properties below is irreflexive :
   (a) $\forall e, e' \in E,\quad [e \neq e',\ {}^\bullet e \cap {}^\bullet e' \neq \emptyset] \ \Rightarrow\ e \# e'$,
   (b) $\forall x, x' \in C \cup E,\quad [\exists e, e' \in E,\ e \# e',\ e \rightarrow^* x,\ e' \rightarrow^* x'] \ \Rightarrow\ x \# x'$.

The change in notations accounts for the usual terminology of *conditions*, instead of places, and *events*, instead of transitions. In an ON, "time" is unfolded, as indicated by (2). A condition can be marked by a unique event (3). By contrast, it may enable several events, which corresponds to a conflict situation. This creates a branching in the net, and the corresponding branches will never meet each other again (4). So conflicts are also unfolded.

ONs are generally introduced to represent runs of a net in the so-called *true concurrency semantics*. To do so, we need extra elements of terminology about ONs. Two nodes $x, x' \in C \cup E$ are said to be *concurrent*, denoted by $x \perp\!\!\!\perp x'$, iff neither $x \# x'$ nor $x \rightarrow^* x'$ nor $x' \rightarrow^* x$ holds. A *co-set* is a set of pairwise concurrent conditions, and a *cut* is a maximal co-set for the inclusion. Finally, a *configuration* is a subset κ of $C \cup E$ which is conflict-free, causally closed (*i.e.* left-closed for $\rightarrow^*$), and such that $\forall e \in E,\ e \in \kappa \Rightarrow e^\bullet \subseteq \kappa$. We denote by $\mathcal{K}_\mathcal{O}$ the set of configurations in $\mathcal{O}$. Observe that conditions form a co-set iff they appear as extremal nodes of a configuration.

Occurrence nets, equipped with morphisms of nets, form the subcategory *Occ* of *Nets*. Observe that in this category, morphisms can only erase (not create) causality or conflict relations between two nodes. Concurrency relations are preserved, and configurations are mapped to configurations.

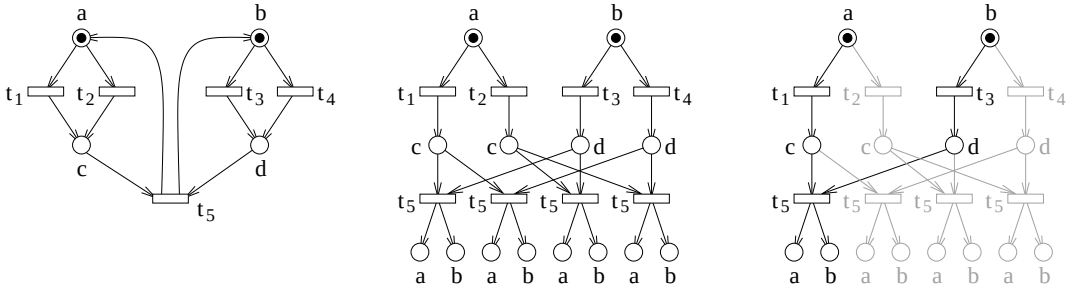

Figure 3: *A net $\mathcal{N}$ (left), a branching process $\mathcal{O}$ of that net (center) and a configuration κ in $\mathcal{O}$ (right), corresponding to a run of $\mathcal{N}$. The folding of $\mathcal{O}$ into $\mathcal{N}$ is represented by transition and place names attached to events and conditions.*

Branching process. $\mathcal{O}$ is said to be a *branching process* (BP) of net $\mathcal{N}$ iff there exists a morphism $\phi : \mathcal{O} \to \mathcal{N}$ satisfying [7]

1. ϕ is a total function on $\mathcal{O}$, also named a *folding* of $\mathcal{O}$,
2. $\forall e, e' \in E,\ [{}^\bullet e = {}^\bullet e',\ \phi(e) = \phi(e')] \ \Rightarrow\ e = e'$.

Notice that being a total function, the restriction $\phi : C^0 \to P^0$ is a bijection, as well as $\phi : {}^\bullet e \to {}^\bullet \phi(e)$ and $\phi : e^\bullet \to \phi(e)^\bullet$ for every event $e \in E$. Formally, and following [7], a branching process is the pair $(\mathcal{O}, \phi)$. Here, with a slight abuse of notation, we often omit mentioning the folding ϕ.

Consider a configuration κ in a branching process $\mathcal{O}$ of $\mathcal{N}$ (see fig. 3). The events of κ are partially ordered by $\to^*$. Let us make a sequence $e_1, e_2, \ldots, e_N$ of these events by taking any linear extension of this partial order. Then $\phi(e_1), \phi(e_2), \ldots, \phi(e_N)$ is a valid run of $\mathcal{N}$. Conversely, provided the BP $\mathcal{O}$ is "large enough,"any run of $\mathcal{N}$ can be recovered in that way from a configuration κ, and the latter is unique by the parsimony condition (2) above.

Unfolding. A *prefix* $\mathcal{O}'$ of an occurrence net $\mathcal{O}$ is defined as a sub-net of $\mathcal{O}$ which is causally closed, contains the initial marking (or equivalently all minimal conditions), and such that $\forall e \in E,\ e \in \mathcal{O}'$ implies $e^\bullet \subseteq \mathcal{O}'$. So a configuration of $\mathcal{O}$ is a conflict free prefix of $\mathcal{O}$. The prefix relation is denoted by $\mathcal{O}' \sqsubseteq \mathcal{O}$.

Given a net $\mathcal{N}$, there exists a maximal branching process of $\mathcal{N}$ for the prefix relation. It is called the *unfolding* of $\mathcal{N}$. We denote it by $\mathcal{U}(\mathcal{N})$, or $\mathcal{U}_\mathcal{N}$ for short, and its corresponding folding by $f_\mathcal{N} : \mathcal{U}_\mathcal{N} \to \mathcal{N}$. Let $\mathcal{O}$ be a branching process of $\mathcal{N}$, with folding $f : \mathcal{O} \to \mathcal{N}$, there exists a unique morphism $\phi : \mathcal{O} \to \mathcal{U}_N$ such that $f = f_\mathcal{N} \circ \phi$. This obvious property on branching processes actually generalizes to any occurrence net $\mathcal{O}$:

$$\forall \mathcal{O} \in Occ\,,\ \forall \phi : \mathcal{O} \to \mathcal{N},\ \exists! \psi : \mathcal{O} \to \mathcal{U}_\mathcal{N},\ \phi = f_\mathcal{N} \circ \psi \tag{1}$$

This *universal property* characterizes the unfolding of $\mathcal{N}$, and can be found in [4].

For the net $\mathcal{N}$ in fig. 3, consider the central branching process $\mathcal{O}$. The unfolding of $\mathcal{N}$ can be obtained by connecting a copy of $\mathcal{O}$ to every pair of conditions (a, b) generated by the same event in $\mathcal{O}$, and so on repeatedly. This example illustrates that an unfolding generally grows in "width" (the conflict dimension) as one progresses in "length" (the time dimension).

There exists a simple and intuitive procedure to build (a prefix of) the unfolding of a net $\mathcal{N}$. We briefly mention it below[5], since it forms the basis of other constructions we use in the sequel.

Procedure 1

- Initialization :
  - Create $|P^0|$ conditions in C^0, and define a bijection $f_\mathcal{N} : C^0 \to P^0$.
  - Set $C = C^0$, $E = \emptyset$ and $\to = \emptyset$.
- Recursion :
  - Let X be a co-set of C and $t \in T$ a transition of $\mathcal{N}$ such that ${}^\bullet t = f_\mathcal{N}(X)$.
  - If there doesn't exist an event e in E with ${}^\bullet e = X$ and $f_\mathcal{N}(e) = t$,
    - create a new event e in E with ${}^\bullet e = X$ and $f_\mathcal{N}(e) = t$,
    - then create a subset Y of $|t^\bullet|$ new conditions in C, set $Y = e^\bullet$, and extend $f_\mathcal{N}$ to have the bijection $f_\mathcal{N} : Y \to t^\bullet$.

[5] This recursive construction is explicit in [4] as well as in [14], where new events and places are named by a backward pointer technique. It also appears in the definition of "canonical branching processes" in [7]. We avoid this heavy notation here, and use instead the less formal "create" primitive.

Product. A product can be defined directly in the category Occ, by the following recursive procedure. Let $\mathcal{O}_1, \mathcal{O}_2$ be two ONs, with $\mathcal{O}_i = (C_i, E_i, \rightarrow_i, C_i^0)$, their product $\mathcal{O} = (C, E, \rightarrow, C^0) = \mathcal{O}_1 \times_O \mathcal{O}_2$ and the associated morphisms $\psi_i : \mathcal{O} \rightarrow \mathcal{O}_i$ are given by :

Procedure 2

- Initialization :
  - Create $|C_1^0| + |C_2^0|$ conditions in C^0, and define injective partial functions $\psi_i : C^0 \rightarrow C_i^0$ in such a way that they have disjoint domains.
  - Set $C = C^0$, $E = \emptyset$ and $\rightarrow = \emptyset$.
- Recursion :
  - Let X be a co-set of C, and $I \subseteq \{1, 2\}$ a non-empty index set ; $\forall i \in I$, let e_i be an event of E_i such that $\psi_i(X) = {}^\bullet e_i$.
  - If there doesn't exist an event $e \in E$ with ${}^\bullet e = X$ and $\forall i \in I,\ \psi_i(e) = e_i$,
    - create a new event e in E with ${}^\bullet e = X$ and $\forall i \in I$, set $\psi_i(e) = e_i$,
    - then create a subset Y of $\sum_{i \in I} |e_i^\bullet|$ new conditions in C, set $e^\bullet = Y$,
    - extend the partial functions ψ_i to Y in order to have disjoint definition domains in Y and to satisfy $\psi_i : Y \rightarrow e_i^\bullet$ injective[6].

The index set I takes value $\{1\}$ or $\{2\}$ to build in $\mathcal{O}$ a "private event" of $\mathcal{O}_1$ or $\mathcal{O}_2$, and takes value $\{1, 2\}$ to build a synchronized event. It can be proved directly that $\times_O$ defined by procedure 2 corresponds to the categorical product of Occ. But this result is derived in a more direct manner in the next sub-section.

2.3 Relations between the two categories

Consider the inclusion functor $F = \subseteq$ of Occ into $Nets$. Following [4], we recall that $G = \mathcal{U}$, the unfolding operation on nets, forms the right adjoint of functor F. To do so, we use the construction of theorem 2-iv, in chap. IV-1 of [1], recalled in appendix A.3, with notations $C{=}Occ$, $D{=}Nets$, $F = \subseteq$ and $G = \mathcal{U}$. The keystone of this construction is the universal property (1) of unfoldings.

Candidate co-unit. The starting point is to find a candidate co-unit $\epsilon : FG \rightarrow \mathbb{I}_D$ for this adjunction, where $\mathbb{I}_D$ is the identity functor in $D{=}Nets$. The co-unit ϵ is defined by a morphism $\epsilon_\mathcal{N}$ for every $\mathcal{N} \in Nets$, $\epsilon_\mathcal{N} : \mathcal{U}_\mathcal{N} \rightarrow \mathcal{N}$. A straightforward choice is the folding $\epsilon_\mathcal{N} = f_\mathcal{N}$. For every $\mathcal{N}$ in $Nets$, we must show that the pair $(G(\mathcal{N}), \epsilon_\mathcal{N}) = (\mathcal{U}_\mathcal{N}, f_\mathcal{N})$ is a universal arrow from F to $\mathcal{N}$ (see (28) in appendix A.3), which is exactly the universal property (1) of the unfolding $\mathcal{U}_\mathcal{N}$. So assumption (UA) holds.

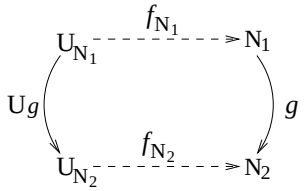

Figure 4: *Commutative diagram satisfied by the unfolding of a morphism.*

[6] on its domain of definition

Unfolding as a functor. As a second step in this construction, we derive that $G = \mathcal{U}$, the unfolding operation on nets, can be turned into a functor from *Nets* to *Occ*. To do so, we must explain how morphisms are transformed by $\mathcal{U}$. Let $g : \mathcal{N}_1 \to \mathcal{N}_2$ be a morphism in *Nets*, $\mathcal{U}(g)$ (or $\mathcal{U}_g$) is defined as the unique morphism from $\mathcal{U}_{\mathcal{N}_1}$ to $\mathcal{U}_{\mathcal{N}_2}$ satisfying (fig. 4)

$$g \circ f_{\mathcal{N}_1} \quad = \quad f_{\mathcal{N}_2} \circ \mathcal{U}(g) \tag{2}$$

Existence and unicity of $\mathcal{U}(g)$ are guaranteed by (1) : take $\mathcal{O} = \mathcal{U}_{\mathcal{N}_1}$, $\mathcal{N} = \mathcal{N}_2$ and $\phi = g \circ f_{\mathcal{N}_1}$. It is then easy to show that $\mathcal{U}$ commutes with the composition of morphisms, and maps identity to identity. Alternatively, it is instructive to build directly $\mathcal{U}(g)$, through a recursion that preserves (2) at each step :

Procedure 3

- Initialization :
  given bijections $f_{\mathcal{N}_i} : C_i^0 \to P_i^0$, $\mathcal{U}_g$ is determined by (2) between C_1^0 and C_2^0.
- Recursion :
  - let X_1 be a co-set of $\mathcal{U}_{\mathcal{N}_1}$ where $\mathcal{U}_g$ is defined, so $g \circ f_{\mathcal{N}_1}(X_1) = f_{\mathcal{N}_2} \circ \mathcal{U}_g(X_1)$, and let e_1 be an event of $\mathcal{U}_{\mathcal{N}_1}$ such that ${}^\bullet e_1 = X_1$,
  - then g is defined on $t_1 = f_{\mathcal{N}_1}(e_1)$ (since it is defined on its pre-set), and ${}^\bullet g(t_1) = g({}^\bullet t_1) = g \circ f_{\mathcal{N}_1}(X_1) = f_{\mathcal{N}_2} \circ \mathcal{U}_g(X_1)$,
  - since $t_2 = g(t_1)$ is enabled by places of $f_{\mathcal{N}_2} \circ \mathcal{U}_g(X_1)$, there exists a unique event e_2 in $\mathcal{U}_{\mathcal{N}_2}$ such that $f_{\mathcal{N}_2}(e_2) = t_2$ and ${}^\bullet e_2 = \mathcal{U}_g(X_1)$, so one must define $\mathcal{U}_g(e_1) = e_2$,
  - given bijections $f_{\mathcal{N}_i} : e_i^\bullet \to t_i^\bullet$, extend $\mathcal{U}_g$ so that it coincides with g between $e_1^\bullet$ and $e_2^\bullet$.

Adjunction. As a last step, we have to show that ϵ is a natural transformation of functor FG into $\mathbb{I}_D$. As FG performs an unfolding in *Nets*, this property coincides with (2) (fig. 4). This is enough to prove the adjunction, *i.e.* to derive a one to one binatural correspondence between morphisms of $\mathrm{Mor}(Occ, Nets)$ and those of $\mathrm{Mor}(Occ, \mathcal{U}(Nets))$, by (30) in appendix A.3.

This particular form of adjunction, where the left adjoint is the inclusion functor, is called a *co-reflection* of the category of occurrence nets in the category of nets. As right adjoints preserve products, one has

$$\mathcal{U}(\mathcal{N}_1 \times_N \mathcal{N}_2) \quad = \quad \mathcal{U}(\mathcal{N}_1) \times_O \mathcal{U}(\mathcal{N}_2) \tag{3}$$

This would actually be sufficient to prove the existence of a product in *Occ* for occurrence nets that are unfoldings. But we can say more. Consider the *unit* of the adjunction, *i.e.* the natural transformation $\eta : \mathbb{I}_D \to GF$, given here by $\forall \mathcal{O} \in Occ$, $\eta_{\mathcal{O}} : \mathcal{O} \to \mathcal{U}_{\mathcal{O}}$. η is obviously a natural equivalence (assumption (NE) of appendix A.4 is satisfied), so the relation between $\times_O$ and $\times_N$ in (3) reaches all elements in *Occ*, which yields the following definition of $\times_O$

$$\mathcal{O}_1 \times_O \mathcal{O}_2 \quad \cong \quad \mathcal{U}(\mathcal{O}_1) \times_O \mathcal{U}(\mathcal{O}_2) \quad = \quad \mathcal{U}(\mathcal{O}_1 \times_N \mathcal{O}_2) \tag{4}$$

where $\cong$ stands for "isomorphic to." The fact that $\times_O$ is a standard product in *Nets* followed by an unfolding gives exactly the recursive definition of the product mentioned at the end of section 2.2 (observe that procedure 2 is indeed built on procedure 1, an unfolding algorithm).

2.4 Multi-clock nets

The notion of trellis that we develop in the sequel only applies to a (large) sub-class of safe Petri nets, that we define now. We show in particular that the categorical constructions developped in section 2.3 adapt to this sub-class. The motivation for this light restriction is discussed later.

Multi-clock net. A *multi-clock net* (MCN) is a tuple $\mathcal{N} = (P, T, \rightarrow, P^0, \nu)$ satisfying;

1. $(P, T, \rightarrow, P^0)$ is an ordinary safe net,
2. $\nu : P \rightarrow P^0$ defines a partition on places, and the restriction $\nu_{|P^0}$ is the identity; we denote by $\bar{p}$ the equivalence class $\nu^{-1}(\nu(p))$ of a place p,
3. $\forall t \in T$, ν is injective on ${}^\bullet t$ and on $t^\bullet$, and $\nu({}^\bullet t) = \nu(t^\bullet)$.

This definition deserves some comments. Observe first that every transition satisfies $|{}^\bullet t| = |t^\bullet|$. So the number of tokens remains constant in a MCN. Moreover, let $M \subseteq P$ be a reachable marking of $\mathcal{N}$, one has $\nu_{|M}$ is bijective. In other words, let $p \in P^0$, at any time there is exactly one place in $\nu^{-1}(p)$ holding a token. Secondly, consider the restriction of $\mathcal{N}$ to places of $\bar{p}$, $p \in P$, that we denote by $\mathcal{N}_{|\bar{p}}$. Then $\mathcal{N}_{|\bar{p}}$ is an automaton, *i.e.* a Petri net where a single place holds a token at any time. Therefore, a multi-clock net can be regarded as a synchronous product of automata (we shall come back on this in section 5, dedicated to labeled nets). By abuse of vocabulary, we will sometimes consider $\bar{p}$ as the state variable of $\mathcal{N}_{|\bar{p}}$ (more rigorously, it corresponds to the value set of this state variable).

Remark: As a graph, $\mathcal{N}_{|\bar{p}}$ has generally several connected components. Only one of them contains the place that is initially marked. Places belonging to the other connected components are unreachable, and can thus be discarded from $\mathcal{N}$ without changing its dynamics. Therefore, in the sequel, we assume that every $\mathcal{N}_{|\bar{p}}$ has a single connected component.

Morphism of MCNs. Let $\mathcal{N}_1, \mathcal{N}_2$ be two MCNs, with $\mathcal{N}_i = (P_i, T_i, \rightarrow_i, P_i^0, \nu_i)$, we restrict ourselves to morphisms $\phi : \mathcal{N}_1 \rightarrow \mathcal{N}_2$ that preserve the partitions ν_1, ν_2, *i.e.* that satisfy

$$\forall p_1 \in P_1,\ \forall p_2 \in P_2, \quad p_1 \,\phi\, p_2 \ \Rightarrow\ \nu_1(p_1)\,\phi\,\nu_2(p_2) \tag{5}$$

The following lemma emphasizes that MCN morphisms actually erase or duplicate state variables as a whole.

Lemma 1 *Let $\phi : \mathcal{N}_1 \rightarrow \mathcal{N}_2$ be a morphism of MCNs, with $\mathcal{N}_i = (P_i, T_i, \rightarrow_i, P_i^0, \nu_i)$.*

a. The inverse image by ϕ of a class of ν_2 is included in a class of ν_1.

b. Given a class of ν_1, ϕ is either defined on all elements of this class, or on none of them.

c. When a place $p_1 \in P_1$ is duplicated by ϕ, i.e. *related to (elements of) several classes of ν_2, each place in $\bar{p_1}$ is duplicated in the same way,* i.e. *related to the same classes.*

Proof. (a) Assume $p_1 \,\phi\, p_2$ and $p_1' \,\phi\, p_2'$ (where $p_i, p_i' \in P_i$), we must show that if $\nu_2(p_2) = \nu_2(p_2')$ then $\nu_1(p_1) = \nu_1(p_1')$. Since ϕ preserves partitions, we have $\nu_1(p_1)\,\phi\,\nu_2(p_2)$ and $\nu_1(p_1)\,\phi\,\nu_2(p_2') = \nu_2(p_2)$ between P_1^0 and P_2^0. But by definition of net morphisms, there is only one place in P_1^0 in relation with $\nu_2(p_2) \in P_2^0$. So $\nu_1(p_1) = \nu_1(p_1')$. Observe that the converse result doesn't hold: One can have $p_1 \,\phi\, p_2$, $p_1' \,\phi\, p_2'$, $\nu_1(p_1) = \nu_1(p_1')$ and $\nu_2(p_2) \neq \nu_2(p_2')$. This is the case in particular when ϕ duplicates some places.

(b) Assume ϕ is defined on $p_1 \in P_1$, with $p_1 \,\phi\, p_2$. Let $t_1 \in p_1^\bullet$, by point 4 in the definition of net morphisms, ϕ is defined on t_1 and one has $\phi(t_1) = t_2 \in p_2^\bullet$. Let p_2' be the unique place in $t_2^\bullet$ such that $\nu_2(p_2') = \nu_2(p_2)$. By point 5 in the definition of net morphisms, there exists a unique p_1' in $t_1^\bullet$ such that $p_1' \,\phi\, p_2'$. Using (a), p_1' is actually the only place in $t_1^\bullet \cap \bar{p_1}$. A similar argument shows that ϕ is also defined on $\bar{p_1} \cap {}^\bullet t_1$ for $t_1 \in {}^\bullet p_1$. By recursion, and since $\mathcal{N}_{1|\bar{p_1}}$ has a single connected component, ϕ is defined on all places of $\bar{p_1}$.

(c) As a by-product of the proof for (b), if $p_1 \phi p_2$, and p_1' satisfies $\nu_1(p_1') = \nu_1(p_1)$, there exists p_2' such that $\nu_2(p_2') = \nu_2(p_2)$ and $p'1\phi p_2'$. □

We define $\overline{Nets}$ as the sub-category of *Nets* formed by multi-clock nets and their morphisms. Notice that it is always possible to turn a safe net $\mathcal{N}$ into a multi-clock net with essentially the same behaviour, simply by adding to each place of $\mathcal{N}$ a complementary place. So $\overline{Nets}$ "almost covers" all *Nets*.

Product. The product $\mathcal{N}_1 \times \mathcal{N}_2$ of two MCNs is defined as the standard product of nets where the resulting partition ν is simply the union of partitions ν_1 and ν_2 (recall that the product builds the disjoint union of places). It is straightforward to check that $\mathcal{N}_1 \times \mathcal{N}_2$ remains a MCN, and that morphisms $\psi_i : \mathcal{N}_1 \times \mathcal{N}_2 \to \mathcal{N}_i$ are morphisms of MCNs. To prove that $\times$ actually defines the categorical product in $\overline{Nets}$, we must check that its universal property holds also in this sub-category. Let $\mathcal{N}$ be a MCN, and let the $f_i : \mathcal{N} \to \mathcal{N}_i$ be morphims of MCNs, $i = 1, 2$. There exists a unique arrow $\phi : \mathcal{N} \to \mathcal{N}_1 \times \mathcal{N}_2$ in *Nets* such that $f_i = \psi_i \circ \phi$. From [6], we know that ϕ is given by $\phi(t) = (f_1(t), f_2(t))$, on transitions where at least one of the f_i is defined. On places, ϕ is given by

$$\begin{array}{lll} p\,\phi\,(p_1, \star) & \text{iff} & p\,f_1\,p_1 \\ p\,\phi\,(\star, p_2) & \text{iff} & p\,f_2\,p_2 \end{array}$$

Therefore ϕ is clearly an arrow of $\overline{Nets}$, and $\times$ defines the categorical product of $\overline{Nets}$.

Multi-clock occurrence nets and unfoldings. A *multi-clock occurrence net* (MCON) is naturally a multi-clock net $\mathcal{O} = (C, E, \to, C^0, \nu)$ where $(C, E, \to, C^0)$ is an occurrence net. They define the sub-category $\overline{Occ}$ of $\overline{Nets}$. The unfolding of a MCN $\mathcal{N}$ is clearly a MCON, and the associated folding $f_{\mathcal{N}} : \mathcal{U}(\mathcal{N}) \to \mathcal{N}$ is of course a morphism in $\overline{Nets}$. So we are in good shape to get a co-reflexion of $\overline{Occ}$ into $\overline{Nets}$. This result can actually be derived exactly as before. One only has to check that the universal property (1) of MCN unfoldings involves MCN morphisms, from which (UA) holds. These simple verifications are left to the reader.

Terminology: In the remaining of the paper, we only deal with multi-clock nets. Therefore the term "multi-clock" will not appear systematically.

3 Trellis nets

This section contains the main contribution of this paper: we introduce trellis nets and study some of their properties. We have insisted in the previous section on known results relating nets to occurrence nets because we will now follow exactly the same track for trellis nets. The next section will then focus on relations between the three notions: nets, trellis nets, and occurrence nets.

3.1 Definition

Pre-trellis net. The (multi-clock) net $\mathcal{T} = (C, E, \to, C^0, \nu)$ is a *pre-trellis net* iff it satisfies:

1. $C^0 = \{c \in C : {}^\bullet c = \emptyset\}$,
2. for every $c \in C^0$, the automaton $\mathcal{T}_{|\bar{c}}$ has no circuit (*i.e.* its flow relation defines a partial order).

The definition of pre-trellis nets is much less restrictive than the definition of occurrence nets. Specifically, point 1 is preserved, point 2 is weakened since $\to^*$ is not any more required to define a partial order, and we have abandoned points 3 and 4: conflicting branches are now allowed to merge on conditions.

As an oriented graph, and by contrast with occurrence nets, a pre-trellis net is not necessarily a partial order. Fig. 5 gives a counter-example of a pre-trellis net containing a circuit. However, one has the following property:

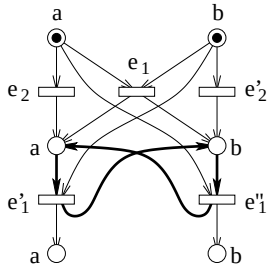

Figure 5: *A pre-trellis net containing a circuit (thick arrows).*

Lemma 2 *No run of a pre-trellis net $\mathcal{T}$ can have a loop,* i.e. *can fill twice the same place. As a consequence, the restriction $\mathcal{T}_{|s}$ of $\mathcal{T}$ to (nodes involved in) any run s defines a partial order of nodes.*

Proof. Assume place c of net $\mathcal{T}$ is filled twice by some sequence s of transitions of $\mathcal{T}$. The canonical projection of s on transitions of $\mathcal{T}_{|\bar{c}}$ is of course a valid run of this automaton. And this sequence fills twice place c, which contradicts the fact that $\mathcal{T}_{|\bar{c}}$ has no circuit. □

Configuration, trellis net. In an occurrence net, every event belongs at least to one configuration, and so is reachable. This is not guaranteed anymore in a pre-trellis net (see $\mathcal{T}_1$ in fig. 7), so we must refine our definition. We define a *configuration* κ of a pre-trellis net $\mathcal{T} = (C, E, \rightarrow, C^0, \nu)$ as a sub-net of $\mathcal{T}$ satisfying

1. $C^0 \subseteq \kappa$,
2. $\forall e \in E \cap \kappa$, ${}^\bullet e \subseteq \kappa$ and $e^\bullet \subseteq \kappa$: each event has all its causes and consequences,
3. $\forall c \in C \cap \kappa$, $|{}^\bullet c \cap \kappa| = 1$ or $c \in C^0$: each condition is either minimal or has one of its possible causes,
4. $\forall c \in C \cap \kappa$, $|c^\bullet \cap \kappa| \leq 1$: each condition triggers at most one event,
5. the restriction of $\mathcal{T}$ to nodes of κ is a partial order.

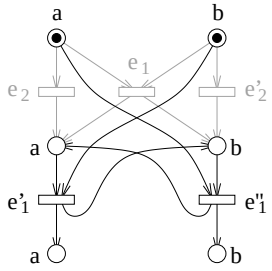

Figure 6: *In the net of fig. 5, a subset of nodes satisfying the first four requirements of a configuration, but failing on the last one.*

This definition is close to the one introduced for ONs, apart from the fact that $|{}^\bullet c| \leq 1$ is not automatic anymore in a pre-trellis net. So one must not only solve conflicts forward (point 4) but also backwards (point 3), to get a valid conflict-free ON. And the requirement that a configuration is "causally closed" is now spread on 2, 3 and 4. The last point is suggested by lemma 2, and is indeed necessary since points 1 to 4 alone do not guarantee this property (see a counter-example in fig. 6). With the above definition, it is straightforward to check that a sequence s is a run of $\mathcal{T}$ iff it corresponds to a linear extension of some configuration κ of $\mathcal{T}$. And so an event of a pre-trellis net is reachable iff it belongs

to a configuration. We thus define a *trellis net* (TN) as a pre-trellis net where each event belongs at least to one finite configuration (see fig. 7 for examples).

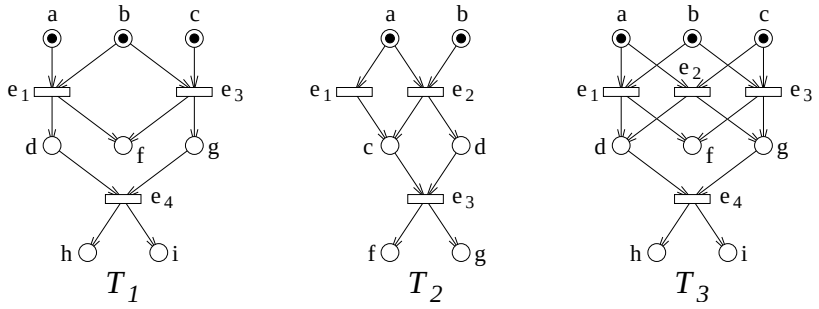

Figure 7: *$\mathcal{T}_1$ is a pre-trellis net but not a trellis net: event e_4 is unreachable. The other nets are trellis nets: all events are reachable. In $\mathcal{T}_2$, e_1 and e_3 are not causally related... but in conflict ! $\mathcal{T}_3$ displays a non binary conflict: $\#(d,f,g)$, but $\perp\!\!\!\perp(d,f)$, $\perp\!\!\!\perp(f,g)$ and $\perp\!\!\!\perp(d,g)$. Removing e_2 in $\mathcal{T}_3$ doesn't yield a valid prefix: we are back to $\mathcal{T}_1$ which is not a trellis net.*

Concurrency and conflict. From the definitions above, one sees that both ONs and TNs are graphical structures encoding families of configurations, in different ways. TNs offer the advantage of being more compact... at the expense of a more complex display of configurations. In particular, the familiar causality, conflict and concurrency relations have no simple graphical characterization (see $\mathcal{T}_2$ in fig. 7). For example, causality is "context dependent:" two events may be concurrent in one configuration, and appear as causally related in another. To define conflict and concurrency, we thus have to abstract the context.

Let $x_1, x_2, \ldots, x_n$ be n nodes of $\mathcal{T}$, they are *concurrent* in $\mathcal{T}$, denoted by $\perp\!\!\!\perp(x_1, x_2, \ldots, x_n)$, iff there exists a configuration κ where they appear as concurrent nodes. The notion of co-set (of conditions) derives from this definition. Observe that concurrency can no longer be derived from pairwise relations, by contrast with ONs (see $\mathcal{T}_3$ in fig. 7). In the same way, $x_1, x_2, \ldots, x_n$ are in conflict, $\#(x_1, x_2, \ldots, x_n)$, iff there is no configuration containing all of them. Again, $\#$ cannot be described by pairwise relations, *i.e.* conflict is not binary in TNs.

Prefix. Prefixes are less easy to define graphically for TNs than for ONs. Let $\mathcal{T}$ be a TN, $\mathcal{T}'$ is a *prefix* of $\mathcal{T}$ ($\mathcal{T}' \sqsubseteq \mathcal{T}$) iff

1. $\mathcal{T}'$ is a sub-net of $\mathcal{T}$,
2. $\{c \text{ condition of } \mathcal{T}, {}^\bullet c = \emptyset\} = \{c' \text{ condition of } \mathcal{T}', {}^\bullet c' = \emptyset\}$
3. $\forall e$ event of $\mathcal{T}$, $\; e \in \mathcal{T}' \;\Rightarrow\; [{}^\bullet e \subseteq \mathcal{T}' \text{ and } e^\bullet \subseteq \mathcal{T}']$,
4. $\mathcal{T}'$ is a trellis net.

The last requirement imposes that every event in the sub-net $\mathcal{T}'$ remains reachable. To illustrate its necessity, consider $\mathcal{T}_3$ in fig. 7: if e_2 is removed, points 1-3 are satisfied, but e_4 becomes unreachable. Of course, $\sqsubseteq$ on TNs extends the relation $\sqsubseteq$ on ONs. Notice also that $\mathcal{T}' \sqsubseteq \mathcal{T}$ implies the existence of an injective morphism $\phi : \mathcal{T}' \to \mathcal{T}$ (which means here that ϕ is a total function).

Height function. The definition of trellis nets now allows to merge conflicting conditions produced by different configurations. However, this leaves a large amount of flexibility to represent a given set of configurations. If one wishes to get a universal object to represent this set, some kind of guideline is necessary to indicate where merges should be performed.

We define a *string* as a configuration $\sigma = (C, E, \rightarrow, C^0, \nu)$ in $\overline{Occ}$ where $|C^0| = 1$. So σ has a single class and thus corresponds to a sequence alternating conditions and events. The *height* $H_\sigma(c)$ of condition c in σ is given by

$$H_\sigma(c) \quad = \quad |\{c' \in C,\ c' \rightarrow^* c\}| \tag{6}$$

In a general configuration κ, we set $H_\kappa(c) \triangleq H_\sigma(c)$ where $\sigma = \kappa_{|\bar{c}}$, so

$$H_\kappa(c) \quad = \quad |\{c' \in C,\ \nu(c') = \nu(c),\ c' \rightarrow^* c\}| \tag{7}$$

A trellis net $\mathcal{T}$ *complies with* H, or is *correctly folded* for H, iff for every condition c of $\mathcal{T}$ and every pair of strings σ, σ' containing c in $\mathcal{T}_{|\bar{c}}$, one has $H_\sigma(c) = H_{\sigma'}(c)$. We denote this common value by $H_\mathcal{T}(c)$ or simply $H(c)$ when there is no ambiguity. Fig. 8 illustrates this property.

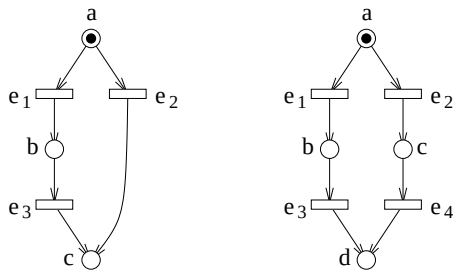

Figure 8: *Two trellis nets; the left one is not H-compliant, the other one is.*

Lemma 3 *The trellis net $\mathcal{T}$ is correctly folded for H iff $\forall c \in C,\ \forall \kappa, \kappa'$ configurations of $\mathcal{T}$ containing c, $H_\kappa(c) = H_{\kappa'}(c)$.*

Proof. This condition is obviously necessary, so we only have to show it is sufficient. Assume $\mathcal{T}$ is not well folded at c, *i.e.* there exists strings σ_1, σ_2 in $\mathcal{T}_{|\bar{c}}$ such that $H_{\sigma_1}(c) \neq H_{\sigma_2}(c)$. WLOG we can assume that $\mathcal{T}$ is correctly folded at all conditions below c in $\mathcal{T}_{|\bar{c}}$. Let $c_1 \rightarrow e_1 \rightarrow c$ in σ_1 and $c_2 \rightarrow e_2 \rightarrow c$ in σ_2. Since e_i is reachable in $\mathcal{T}$, there exists a configuration κ_i containing c_i, e_i and c in $\mathcal{T}$. One has $H_{\sigma_i}(c_i) = H_{\kappa_i}(c_i)$, because $\mathcal{T}$ is correctly folded at c_i. Adding 1 to both sides of the equality, we get $H_{\sigma_i}(c) = H_{\kappa_i}(c)$. So $H_{\sigma_1}(c) \neq H_{\sigma_2}(c)$ entails $H_{\kappa_1}(c) \neq H_{\kappa_2}(c)$. □

In the sequel, we only consider H-compliant TNs. The latter, associated to the usual notion of morphism (of MCNs), form the category $\overline{Tr}$. So we have three nested categories: $\overline{Occ} \subset \overline{Tr} \subset \overline{Nets}$.

3.2 Trellis processes and time unfoldings

Trellis Process. Following ideas developed for occurrence nets, trellis nets can be used to represent runs of a given net. The trellis net $\mathcal{T} = (C, E, \rightarrow, C^0, \nu)$ is a *trellis process* (TP) of net $\mathcal{N}$ iff there exists a morphism $\phi : \mathcal{T} \rightarrow \mathcal{N}$ satisfying

1. ϕ is a folding of $\mathcal{T}$ (*i.e.* a total function on $\mathcal{T}$),
2. $\forall e, e' \in E,\ [{}^\bullet e = {}^\bullet e',\ \phi(e) = \phi(e')] \ \Rightarrow\ e = e'$,
3. $\forall c, c' \in C,\ [H(c) = H(c'),\ \phi(c) = \phi(c')] \ \Rightarrow\ c = c'$.

Notice that being a total function, the restriction $\phi : C^0 \rightarrow P^0$ is a bijection, as well as $\phi : {}^\bullet e \rightarrow {}^\bullet \phi(e)$ and $\phi : e^\bullet \rightarrow \phi(e)^\bullet$ for every event $e \in E$. By contrast with branching processes, this definition contains a double parsimony condition: by 2, redundant branchings are eliminated, and by 3, merges are imposed. Again, a trellis process is formally the pair $(\mathcal{T}, \phi)$, but we will often omit mentioning ϕ.

By definition ϕ is a folding of $\mathcal{T}$ into $\mathcal{N}$, so every configuration κ of $\mathcal{T}$ represents a run of $\mathcal{N}$ in the true concurrency semantics, and has a counterpart in $\mathcal{U}_\mathcal{N}$. So a trellis process of $\mathcal{N}$ corresponds to a collection of runs of $\mathcal{N}$. Conversely, a run of $\mathcal{N}$ is represented by at most one configuration in $\mathcal{T}$: If κ_1 and κ_2 are isomorphic and folded to $\mathcal{N}$ in the same way (up to this isomorphism), then they are identical. Indeed, one has $H_{\kappa_1} = H_{\kappa_2}$ which shows that conditions are identical (point 3), from which events are also identical (point 2).

Let us consider the restriction of a TP $\mathcal{T}$ of $\mathcal{N}$ to nodes with height lower than $h \in \mathbb{N}$. The two remarks above indicate that this restriction is a finite TN, since the restriction of $\mathcal{U}_\mathcal{N}$ to nodes lower than h is itself finite. This property opens the way to recursive reasonings on trellis processes.

Time unfolding of a net. As for branching processes, one can easily build trellis processes of a net $\mathcal{N} = (P, T, \rightarrow, P^0, \mu)$ with a simple recursion, yielding both $\mathcal{T} = (C, E, \rightarrow, C^0, \nu)$ and the folding $\phi : \mathcal{T} \rightarrow \mathcal{N}$.

Procedure 4

- Initialization :
  - Create $|P^0|$ conditions in C^0, and define a bijection $\phi : C^0 \rightarrow P^0$.
  - Set $C = C^0$, $E = \emptyset$, $\rightarrow = \emptyset$ and $\nu = Id$.
- Recursion :
  - Let X be a co-set of C and $t \in T$ a transition of $\mathcal{N}$ such that ${}^\bullet t = \phi(X)$.
  - If there doesn't exist an event $e \in E$ with ${}^\bullet e = X$ and $\phi(e) = t$,
    - create a new event e in E with ${}^\bullet e = X$ and $\phi(e) = t$,
    - create a subset Y of $|t^\bullet|$ new conditions in C, with $Y = e^\bullet$,
      extend ϕ to have $\phi : Y \rightarrow t^\bullet$ bijective,
      and extend the partition ν to preserve $\nu = \phi^{-1} \circ \mu$,
    - then, for every $c \in Y$,
      if $\exists c' \in C$, $\phi(c') = \phi(c)$ and $H(c') = H(c)$ then merge c and c'.

Procedure 4 is a variation of procedure 1 (that builds BPs of a net $\mathcal{N}$). It essentially differs by the last steps, where the newly created conditions are merged to existing ones as soon as they represent the same place and have the same height.

Procedure 4 generates a trellis process of $\mathcal{N}$, by construction, and conversely, it is easily checked that *any* TP of $\mathcal{N}$ can be reached in that way. The proof relies on two facts : First, as soon as a subset of conditions X is declared as "co-set," this property remains true forever, whatever the next steps of the recursion are. And secondly, every event e in a TP is reachable, *i.e.* belongs to a configuration. So the connection of an event e to its co-set X of pre-conditions can't be prevented in the procedure, whatever the ordering of operations.

As for unfoldings, we would like to prove the existence of a unique stationary point of procedure 4. We start by defining the union of TPs. Let $\mathcal{T}_1, \mathcal{T}_2$ be TPs of $\mathcal{N}$, with respective foldings ϕ_i. The union $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$, its folding ϕ and morphisms $\psi_i : \mathcal{T}_i \rightarrow \mathcal{T}$ are defined by the following

Procedure 5

- Initialization :
  - Create $|P^0|$ conditions in C^0, define a bijection $\phi : C^0 \rightarrow P^0$, and bijections $\psi_i : C^0 \rightarrow C_i^0$ satisfying $\phi_i = \phi \circ \psi_i$, $i \in \{1, 2\}$.
  - Set $C = C^0$, $E = \emptyset$, $\rightarrow = \emptyset$ and $\nu = Id$.

- Recursion :
  - for $i \in \{1,2\}$ and $e_i \in E_i$ such that ψ_i is defined on ${}^\bullet e_i$ but not on e_i
  - if $\exists\, e \in E$ such that ${}^\bullet e = \psi_i({}^\bullet e_i)$ and $\phi(e) = \phi_i(e_i)$, set $\psi_i(e_i) = e$, and define $\psi_i : e_i^\bullet \to e^\bullet$ in order to preserve $\phi_i = \phi \circ \psi_i$,
  - otherwise create a new $e \in E$ with ${}^\bullet e = \psi_i({}^\bullet e_i),\ \phi(e) = \phi_i(e_i),\ \psi_i(e_i) = e$,
  - then, for every condition $c_i \in e_i^\bullet$
    - if $\exists\, c \in C,\ H(c) = H(c_i)$ and $\phi(c) = \phi_i(c_i)$, then add e in ${}^\bullet c$ and set $\psi_i(c_i) = c$,
    - otherwise create a new $c \in C$ with ${}^\bullet c = e,\ \phi(c) = \phi_i(c_i),\ \psi_i(c_i) = c$, and set $\nu(c) = \psi_i \circ \nu_i(c_i)$.

Clearly, procedure 5 yields a TP of $\mathcal{N}$, a folding $\phi : \mathcal{T} \to \mathcal{N}$, and morphisms $\psi_i : \mathcal{T}_i \to \mathcal{T}$. The latter are injective total functions, which proves $\mathcal{T}_i \sqsubseteq \mathcal{T}$, and $\mathcal{T}$ is the smallest TP of $\mathcal{N}$ having $\mathcal{T}_1, \mathcal{T}_2$ as prefixes. Using this property, one has that a TP of $\mathcal{N}$ is isomorphic to the union of its configurations, or conversely, that a set of configurations determines a unique TP. Notice also that there exist unique morphisms $\psi_i : \mathcal{T}_i \to \mathcal{T}_1 \cup \mathcal{T}_2$ satisfying $\phi_i = \phi \circ \psi_i$: precisely the ones obtained by procedure 5.

Procedure 5 generalizes without difficulty to the union of an arbitrary number of trellis processes[7] of $\mathcal{N}$.

The intersection $\mathcal{T}_1 \cap \mathcal{T}_2$ can be defined in a similar manner, by a simple modification of procedure 5, or by taking the union of configurations in $\mathcal{K}$

$$\mathcal{K} \;=\; \{\kappa_1 \in \mathcal{K}_{\mathcal{T}_1} \,:\, \exists \kappa_2 \in \mathcal{K}_{\mathcal{T}_2},\ \exists\, \phi : \kappa_1 \to \kappa_2 \ \text{ isomorphism},\ \phi_1 = \phi_2 \circ \phi\} \tag{8}$$

Theorem 1 *Let $\mathcal{N} \in \overline{Nets}$, there exists a unique maximal trellis process of $\mathcal{N}$ for the prefix relation. We call it the* trellis *of $\mathcal{N}$ or the* time unfolding *of $\mathcal{N}$, and denote it by $\mathcal{U}^t_{\mathcal{N}}$, with corresponding folding $f^t_{\mathcal{N}} : \mathcal{U}^t_{\mathcal{N}} \to \mathcal{N}$.*

$\mathcal{U}(\mathcal{N})$, the unfolding of $\mathcal{N}$, and $\mathcal{U}^t(\mathcal{N})$, the time-unfolding of $\mathcal{N}$, are different encodings for the set of trajectories of $\mathcal{N}$. In particular, they have the same configuration set.

Proof. We have proved that the collection of TPs of $\mathcal{N}$ is stable by arbitrary union, and that $\mathcal{T} \sqsubseteq \mathcal{T} \cup \mathcal{T}'$. So there exists a unique maximal TP of $\mathcal{N}$ for the prefix relation. Moreover, since every finite TP of $\mathcal{N}$ is reachable by procedure 4, this procedure converges to $\mathcal{U}^t_{\mathcal{N}}$, its unique stationary point.

Given a net $\mathcal{N}$, consider $\mathcal{K}$, the set of all configurations in $\mathcal{U}_{\mathcal{N}}$. Every configuration of $\mathcal{K}$ is a TP of $\mathcal{N}$, and every configuration in a TP of $\mathcal{N}$ is in $\mathcal{K}$. So $\mathcal{U}^t_{\mathcal{N}}$, the trellis of $\mathcal{N}$, is the union in the sense of trellis processes of configurations of $\mathcal{K}$, just like the unfolding $\mathcal{U}_{\mathcal{N}}$ can be viewed as the union in the sense of branching processes of all configurations of $\mathcal{K}$. As a consequence, $\mathcal{U}_{\mathcal{N}}$ and $\mathcal{U}^t_{\mathcal{N}}$ describe the same set of configurations. $\mathcal{U}^t_{\mathcal{N}}$ can actually be recovered from $\mathcal{U}_{\mathcal{N}}$ by a folding operation : merge conditions c, c' such that $f_{\mathcal{N}}(c) = f_{\mathcal{N}}(c')$ and $H(c) = H(c')$, in order to ensure point 3 in the definition of a TP, then merge redundant events violating the parsimony condition (point 2). We shall express this relation more formally in section 4. □

Theorem 2 (Universal property of $\mathcal{U}^t_{\mathcal{N}}$) *Let $\mathcal{N} \in \overline{Nets}$, for every trellis net $\mathcal{T}$ in $\overline{Tr}$ and morphism $\phi : \mathcal{T} \to \mathcal{N}$, there exists a unique morphism $\psi : \mathcal{T} \to \mathcal{U}^t_{\mathcal{N}}$ such that $\phi = f^t_{\mathcal{N}} \circ \psi$.*

[7] For branching processes, Engelfriet [7] proceeds by isolating a canonical representent for a class of isomorphic branching processes, *i.e.* BPs formed by the same occurrence net but different foldings, while still representing the same configurations. The union is then defined on these canonical BPs. Here, we circumvent this construction by handling equivalence classes "as a whole." Adding an equivalent BP in the union doesn't change the result.

Proof. We proceed in several steps. If ψ exists, ϕ and ψ necessarily have identical domains of definition. So let $\mathcal{T}' = \mathcal{T}_{|dom(\phi)}$, and let π be the canonical projection from $\mathcal{T}$ to $\mathcal{T}'$. By lemma 1, $\mathcal{T}'$ is the restriction of $\mathcal{T}$ to a subset of its state variables, so by definition of H, $\mathcal{T}'$ remains a correctly folded trellis net. There exists a unique $\phi' : \mathcal{T}' \to \mathcal{N}$ such that $\phi = \phi' \circ \pi$, and if ψ exists, there exists as well a unique $\psi' : \mathcal{T}' \to \mathcal{U}^t_{\mathcal{N}}$ such that $\psi = \psi' \circ \pi$. The relation $\phi = f^t_{\mathcal{N}} \circ \psi$ entails $\phi' \circ \pi = f^t_{\mathcal{N}} \circ \psi' \circ \pi$, and since π is obviously an epi-morphism, we get $\phi' = f^t_{\mathcal{N}} \circ \psi'$. So we can simplify the problem and assume that ϕ is defined everywhere on $\mathcal{T}$.

If $\mathcal{T}$ is a trellis process of $\mathcal{N}$, the existence and uniqueness of ψ derives directly from procedure 5 applied to $\mathcal{T}_1 = \mathcal{T}$ and $\mathcal{T}_2 = \mathcal{U}^t_{\mathcal{N}}$. In the case of a general trellis net $\mathcal{T}$, we proceed in a similar manner and build a morphism $\psi : \mathcal{T} \to \mathcal{U}^t_{\mathcal{N}}$ recursively on events of $\mathcal{T}$ (recall that events of $\mathcal{T}$ are all reachable). We adopt the notation $\mathcal{U}^t_{\mathcal{N}} = (C', E', \to', C'^0, \nu')$.

As a start point, we define ψ between C^0 and C'^0. Since the restriction $f^t_{\mathcal{N}} : C'^0 \to P^0$ is bijective, we must take

$$\forall c \in C^0,\ \forall c' \in C'^0, \qquad c \,\psi\, c' \quad \Leftrightarrow \quad c \,\phi\, f^t_{\mathcal{N}}(c') \tag{9}$$

We design the following steps in order to ensure that ψ, *restricted to its current domain of definition,* remains a morphism (of MCNs) and satisfies $\phi = f^t_{\mathcal{N}} \circ \psi$. By (9), this is obviously true at the start point. We show that this property can be progressively extended to cover all $\mathcal{T}$, and thus satisfy point 3 in the definition of a morphism.

Consider an event e of $\mathcal{T}$, and assume ψ is defined on the co-set $X = {}^\bullet e$ but not at e. ψ is a morphism (on its domain of definition) so $\psi(X)$[8] is a co-set of $\mathcal{U}^t_{\mathcal{N}}$. Since ${}^\bullet\phi(e) = \phi(X)$ in $\mathcal{N}$, there exists a unique event e' in $\mathcal{U}^t_{\mathcal{N}}$ such that ${}^\bullet e' = \psi(X)$ and $f^t_{\mathcal{N}}(e') = \phi(e)$. To preserve $\phi = f^t_{\mathcal{N}} \circ \psi$, we must extend ψ by $\psi(e) = e'$, and since $f^t_{\mathcal{N}} : e'^\bullet \to \phi(e)^\bullet$ is bijective, the extension of ψ between $e^\bullet$ and $e'^\bullet$ is also imposed, as in (9). By construction, this extended ψ satisfies points 1, 2 and 4 of the definition of net morphisms. And point 3 is obtained when we restrict ψ to its domain of definition. Finally, as a relation on conditions and relying on (9), ψ clearly preserves partitions ν and ν'.

Since all events of $\mathcal{T}$ are reachable, ψ can be extended to finally cover all $\mathcal{T}$. This proves both existence and uniqueness of ψ. □

3.3 Co-reflection of $\overline{Tr}$ into $\overline{Nets}$

To match again notations of appendix A, define categories $C = \overline{Tr}$ and $D = \overline{Nets}$, and take for functor $F : C \to D$ the inclusion functor. We prove that the time unfolding operation on nets can be turned into a functor $G = \mathcal{U}^t : D \to C$, *i.e.* $\mathcal{U}^t : \overline{Nets} \to \overline{Tr}$, which is the right adjoint of F. We proceed as in section 2.3 : the derivation of the adjunction is based on the universal property stated in theorem 2.

Candidate co-unit. As for unfoldings, we look first for a candidate co-unit $\epsilon : FG \to \mathbb{I}_D$ for this adjunction. The co-unit ϵ is defined as a collection of morphisms $\epsilon_{\mathcal{N}} : \mathcal{U}^t_{\mathcal{N}} \to \mathcal{N}$, for every $\mathcal{N} \in \overline{Nets}$. An obvious choice is the folding $\epsilon_{\mathcal{N}} = f^t_{\mathcal{N}}$. For every $\mathcal{N} \in \overline{Nets}$, we must show that the pair $(G(\mathcal{N}), \epsilon_{\mathcal{N}}) = (\mathcal{U}^t_{\mathcal{N}}, f^t_{\mathcal{N}})$ is a universal arrow from F to $\mathcal{N}$ (see (28) in appendix A.3), which is exactly the universal property of the time unfolding $\mathcal{U}^t_{\mathcal{N}}$ expressed in theorem 2. So assumption (UA) holds.

Time unfolding as a functor. Let $g : \mathcal{N}_1 \to \mathcal{N}_2$ be a morphism in $\overline{Nets}$, $\mathcal{U}^t(g)$ can be defined as the unique morphism in $\overline{Tr}$ satisfying :

$$g \circ f^t_{\mathcal{N}_1} \quad = \quad f^t_{\mathcal{N}_2} \circ \mathcal{U}^t(g) \tag{10}$$

[8] By abuse of notation, we write $\psi(X)$ the set $\{c' \in C' \,:\, \exists c \in X,\, c \,\psi\, c'\}$.

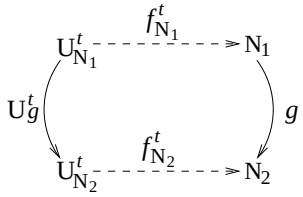

Figure 9: *Commutative diagram satisfied by the time unfolding of a morphism.*

Existence and unicity of $\mathcal{U}^t(g)$ are guaranteed by theorem 2, and (10) is sufficient to prove that $\mathcal{U}^t$ is indeed a functor. Moreover, in practice, it is possible to define $\mathcal{U}^t(g)$ recursively with procedure 3, where $\mathcal{U}_g$ is replaced by $\mathcal{U}^t_g$, $\mathcal{U}_{\mathcal{N}_i}$ by $\mathcal{U}^t_{\mathcal{N}_i}$, the folding $f_{\mathcal{N}_i}$ by $f^t_{\mathcal{N}_i}$, and where the invariant (2) is replaced by (10).

Co-reflection. Finally, with a now classical argument, (10) reveals that ϵ is actually a natural transformation of functor FG into $\mathbb{I}_D$, which allows to derive the one to one binatural correspondence between morphisms of $\mathrm{Mor}(\overline{Tr}, \overline{Nets})$ and those of $\mathrm{Mor}(\overline{Tr}, \mathcal{U}^t(\overline{Nets}))$, by (30) in appendix A.3. This evidences the co-reflection of $\overline{Tr}$ into $\overline{Nets}$.

Product in $\overline{Tr}$. An important property we expect from trellis processes concerns their factorization. Indeed, this property forms the heart of modular/distributed algorithms, that we have based on unfoldings up to now [23]. The co-reflection of the category of trellis nets into the category of nets yields directly this factorization property.

As right adjoints preserve products, one has

$$\mathcal{U}^t(\mathcal{N}_1 \times_{\overline{N}} \mathcal{N}_2) \quad = \quad \mathcal{U}^t(\mathcal{N}_1) \times_{\overline{T}} \mathcal{U}^t(\mathcal{N}_2) \tag{11}$$

which also proves the existence of a product in $\overline{Tr}$ for trellis nets that are time unfoldings of a net. But the unit η of the adjunction, *i.e.* the natural transformation $\eta : \mathbb{I}_D \to GF$ defined here by $\eta_{\mathcal{T}} : \mathcal{T} \to \mathcal{U}^t_{\mathcal{T}}$ for every $\mathcal{T}$ in $\overline{Tr}$, is obviously a natural equivalence ($\eta_{\mathcal{T}}$ is the identity). In other words, assumption (NE) of appendix A.4 is satisfied, so the product $\times_{\overline{N}}$ reaches all elements in $\overline{Tr}$. We can thus *define* the product $\times_{\overline{T}}$ in $\overline{Tr}$ by

$$\mathcal{T}_1 \times_{\overline{T}} \mathcal{T}_2 \quad \cong \quad \mathcal{U}^t(\mathcal{T}_1) \times_{\overline{T}} \mathcal{U}^t(\mathcal{T}_2) \quad = \quad \mathcal{U}^t(\mathcal{T}_1 \times_{\overline{N}} \mathcal{T}_2) \tag{12}$$

As for $\times_{\overline{O}}$, the fact that $\times_{\overline{T}}$ is a standard product in $\overline{Nets}$ followed by a time unfolding provides a recursive definition of the product in $\overline{Tr}$, based on procedure 4. Let $\mathcal{T}_1, \mathcal{T}_2$ be two TNs, with $\mathcal{T}_i = (C_i, E_i, \to_i, C^0_i, \nu_i)$, their product $\mathcal{T} = (C, E, \to, C^0, \nu) = \mathcal{T}_1 \times_{\overline{T}} \mathcal{T}_2$ and the associated morphisms $\psi_i : \mathcal{T} \to \mathcal{T}_i$ are given by :

Procedure 6

- Initialization :
  - Create $|C^0_1| + |C^0_2|$ conditions in C^0, and define injective partial functions $\psi_i : C^0 \to C^0_i$ in such a way that they have disjoint domains.
  - Set $C = C^0$, $E = \emptyset$, $\to = \emptyset$ and $\nu = Id$.
- Recursion :
  - Let X be a co-set of C, and $I \subseteq \{1, 2\}$ a non-empty index set ;
    $\forall i \in I$, let e_i be an event of E_i such that $\psi_i(X) = {}^\bullet e_i$.
  - If there doesn't exist an event $e \in E$ with ${}^\bullet e = X$ and $\forall i \in I,\ \psi_i(e) = e_i$,

* create a new event $e \in E$ with ${}^\bullet e = X$, and $\forall i \in I$ set $\psi_i(e) = e_i$,
* create a subset Y of $\sum_{i\in I} |e_i^\bullet| = |X|$ new conditions in C, set $e^\bullet = Y$,
* extend the partial functions $\psi_i,\ i \in I$, to Y in order to have disjoint definition domains in Y and to satisfy $\psi_i : Y \to e_i^\bullet$ injective[9],
* $\forall c \in Y$ define $\nu(c)$ as $\psi_i^{-1} \circ \nu_i \circ \psi_i(c)$ for the unique ψ_i defined at c,
* $\forall i \in I,\ \forall c \in Y$, if $\exists c' \in C,\ \psi_i(c') = \psi_i(c)$ and $H(c') = H(c)$ then merge conditions c and c'.

4 Relations between nets, trellises and unfoldings

4.1 Co-reflection of $\overline{Occ}$ into $\overline{Tr}$

At this point, we have three nested categories $\overline{Occ} \subset \overline{Tr} \subset \overline{Nets}$. By restricting $\overline{Nets}$ to $\overline{Tr}$ in the co-reflection of $\overline{Occ}$ into $\overline{Nets}$, we can derive another adjunction between $\overline{Occ}$ and $\overline{Tr}$ (actually another co-reflection). Specifically, take categories $C = \overline{Occ}$ and $D = \overline{Tr}$, with the inclusion functor for $F : \overline{Occ} \to \overline{Tr}$ and the unfolding functor for $G : \overline{Tr} \to \overline{Occ}$. Notice that applying $\mathcal{U}$ to a trellis net $\mathcal{T}$ performs an unfolding in the "conflict dimension," since time is already unfolded (each $\mathcal{T}_{|\bar{c}}$ is a partial order of nodes). We thus denote by $G = \mathcal{U}^c$ the restriction of $\mathcal{U}$ to $\overline{Tr}$.

In this adjunction, the universal property of "conflict unfoldings," corresponding to assumption (UA), yields

$$\forall \mathcal{T} \in \overline{Tr},\ \forall \mathcal{O} \in \overline{Occ},\ \forall \phi : \mathcal{O} \to \mathcal{T},\ \ \exists! \psi : \mathcal{O} \to \mathcal{U}^c_{\mathcal{T}},\ \ \phi = f^c_{\mathcal{T}} \circ \psi \tag{13}$$

where $f^c_{\mathcal{T}} : \mathcal{U}^c_{\mathcal{T}} \to \mathcal{T}$ is the folding of $\mathcal{U}^c(\mathcal{T})$ into $\mathcal{T}$. Let $\mathcal{T}_1, \mathcal{T}_2$ be two TNs, the limit preservation theorem on right adjoints gives:

$$\mathcal{U}^c(\mathcal{T}_1 \times_{\overline{T}} \mathcal{T}_2) \quad = \quad \mathcal{U}^c(\mathcal{T}_1) \times_{\overline{O}} \mathcal{U}^c(\mathcal{T}_2) \tag{14}$$

And finally, using the fact that the unit of the adjunction, $\eta_{\mathcal{O}} : \mathcal{O} \to \mathcal{U}^c(\mathcal{O})$, defines a natural equivalence in $\overline{Occ}$ (assumption (NE)), we can actually use this relation to (re)define the product $\times_{\overline{O}}$ by

$$\mathcal{O}_1 \times_{\overline{O}} \mathcal{O}_2 \ \cong \ \mathcal{U}^c(\mathcal{O}_1) \times_{\overline{O}} \mathcal{U}^c(\mathcal{O}_2) \quad = \quad \mathcal{U}^c(\mathcal{O}_1 \times_{\overline{T}} \mathcal{O}_2) \tag{15}$$

4.2 Composition of adjunctions

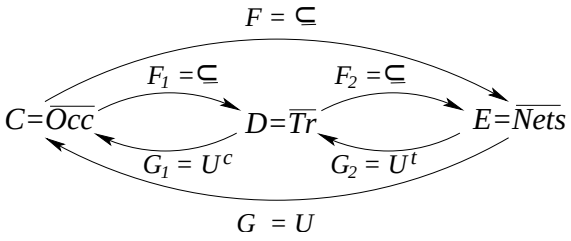

Figure 10: *Adjunctions relating categories* $\overline{Occ}$, $\overline{Tr}$ *and* $\overline{Nets}$.

Gathering results obtained so far, we have three adjunctions relating categories $C = \overline{Occ}$, $D = \overline{Tr}$ and $E = \overline{Nets}$, as displayed by figure 10. It is a well known fact that adjunctions can be composed ([1], chap. IV-8, thm 1), so $F_2 \circ F_1 : \overline{Occ} \to \overline{Nets}$ and $G_1 \circ G_2 : \overline{Nets} \to \overline{Occ}$ defines an adjunction.

[9] on its domain of definition

Since $F_2 \circ F_1 = F$ is the inclusion functor, we thus have that $G = G_1 \circ G_2$, up to a natural equivalence[10]. This translates into

$$\forall \mathcal{N} \in \overline{Nets}, \quad \mathcal{U}(\mathcal{N}) \quad \cong \quad \mathcal{U}^c \circ \mathcal{U}^t(\mathcal{N}) \tag{16}$$

and naturally the corresponding foldings can be composed: $f_{\mathcal{N}} = f^t_{\mathcal{N}} \circ f^c_{\mathcal{U}^t_{\mathcal{N}}}$, up to the isomorphism in (16).

Notice that (16) expresses that the time-unfolding $\mathcal{U}^t(\mathcal{N})$ of a net can be recovered by "refolding" conflicts on its full unfolding $\mathcal{U}_{\mathcal{N}}$. Specifically, $f^c_{\mathcal{U}^t_{\mathcal{N}}} : \mathcal{U}(\mathcal{N}) \to \mathcal{U}^t(\mathcal{N})$ merges conditions with the same height and representing the same place of $\mathcal{N}$, then merges (or removes) redundant events representing the same transition connected to a given co-set. This is illustrated in fig. 11 that compares the unfolding and the trellis of a net.

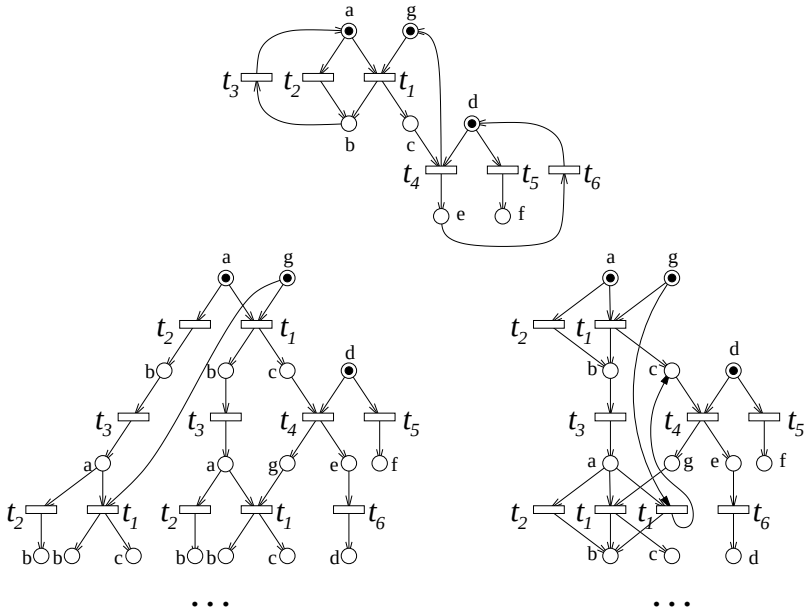

Figure 11: *A net $\mathcal{N}$ (top), with 3 sequential components, defined by $\{a, b\}, \{g, c\}$ and $\{d, e, f\}$. Its unfolding $\mathcal{U}_{\mathcal{N}}$ (bottom left), and its trellis $\mathcal{U}^t_{\mathcal{N}}$ (bottom right). For clarity, conditions/events are labeled by the place/transition they represent in $\mathcal{N}$, instead of having distinct names.*

In terms of product preservation, the composition of adjoints yields, for any pair $\mathcal{N}_1, \mathcal{N}_2$ in $\overline{Nets}$

$$\begin{aligned}
\mathcal{U}(\mathcal{N}_1 \times_{\overline{N}} \mathcal{N}_2) &= \mathcal{U}(\mathcal{N}_1) \times_{\overline{O}} \mathcal{U}(\mathcal{N}_2) && (17)\\
&\cong \mathcal{U}^c \circ \mathcal{U}^t(\mathcal{N}_1 \times_{\overline{N}} \mathcal{N}_2) && (18)\\
&= \mathcal{U}^c(\, \mathcal{U}^t(\mathcal{N}_1) \times_{\overline{T}} \mathcal{U}^t(\mathcal{N}_2)\,) && (19)\\
&= \mathcal{U}^c(\mathcal{U}^t(\mathcal{N}_1)) \times_{\overline{O}} \mathcal{U}^c(\mathcal{U}^t(\mathcal{N}_2)) && (20)
\end{aligned}$$

5 Application to labeled nets

The standard product of nets is generally not used in its basic form, but is rather constrained by a *synchronization algebra*. The latter specifies where synchronizations must take place, and what transitions can be considered as "private" to a component. We recall this notion below, in order to relate more precisely the time unfolding of a net to the standard notion of trellis of an automaton.

[10]The adjoint of a functor is unique up to a natural equivalence, see [1], chap. IV-1, cor. 1.

5.1 Categories of labeled nets

A (multi-clock) *labeled net* $\mathcal{N} = (P, T, \rightarrow, P^0, \mu, \lambda, \Lambda)$ is a net extended with a labeling function λ on T, taking values in the alphabet Λ. As an illustration, we choose a simple synchronization algebra on labels : given two components, the intersection of their label sets defines the "common letters," and

- a transition labeled by a common letter must synchronize with a transition carrying the same label in the other component,
- a transition labeled by a private letter remains private.

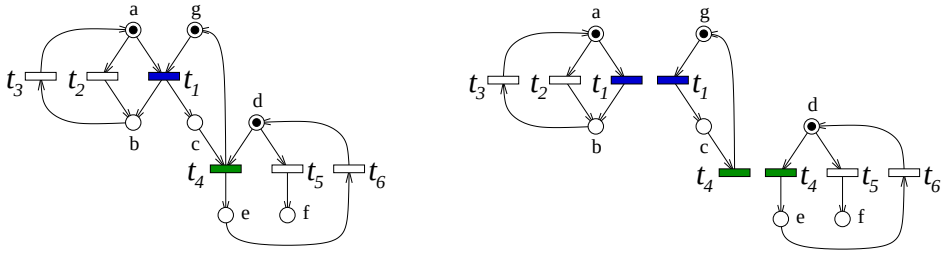

Figure 12: *A multi-clock net $\mathcal{N}$ (left) and its decomposition as a product of labeled automata $\mathcal{N}_{|\bar{a}}, \mathcal{N}_{|\bar{g}}, \mathcal{N}_{|\bar{d}}$ (right). The labels are transition names.*

Based on this synchronization algebra, the product of nets takes the following form. Let $\mathcal{N}_1, \mathcal{N}_2$ be two labeled nets, $\mathcal{N}_i = (P_i, T_i, \rightarrow_i, P_i^0, \mu_i, \lambda_i, \Lambda_i)$, their product is defined as the net $\mathcal{N} = (P, T, \rightarrow, P^0, \mu, \lambda, \Lambda)$ (and associated morphisms $\psi_i : \mathcal{N} \rightarrow \mathcal{N}_i$) satisfying

1. $P = \{(p_1, \star) : p_1 \in P_1\} \cup \{(\star, p_2) : p_2 \in P_2\}$, where $\star$ means "empty,"
   $\psi_1(p_1, p_2) = p_1$ if $p_1 \neq \star$ and is undefined otherwise (symmetrically for ψ_2),
2. $P^0 = \psi_1^{-1}(P_1^0) \cup \psi_2^{-1}(P_2^0)$,
3. $T = \{(t_1, \star) : \ t_1 \in T_1, \ \lambda_1(t_1) \in \Lambda_1 \setminus \Lambda_2\}$
   $\cup \{(t_1, t_2) : \ t_1 \in T_1, t_2 \in T_2, \lambda_1(t_1) = \lambda_2(t_2) \in \Lambda_1 \cap \Lambda_2\}$
   $\cup \{(\star, t_2) : t_2 \in T_2, \lambda_2(t_2) \in \Lambda_2 \setminus \Lambda_1\}$,
   $\psi_1(t_1, t_2) = t_1$ if $t_1 \neq \star$ and is undefined otherwise (symmetrically for ψ_2),
4. $\rightarrow$ is defined by ${}^\bullet t = {}^\bullet\psi_1(t) \cup {}^\bullet\psi_2(t)$ and $t^\bullet = \psi_1(t)^\bullet \cup \psi_2(t)^\bullet$, assuming ${}^\bullet\psi_i(t) = \psi_i(t)^\bullet = \emptyset$ if ψ_i is undefined on t,
5. $\Lambda = \Lambda_1 \cup \Lambda_2$; λ is the obvious and unique labeling that makes ψ_1, ψ_2 label preserving morphisms ; μ is the disjoint union of partitions μ_1, μ_2.

Observe that this product is a restriction of the product in $\overline{Nets}$, where transition pairs not satisfying the synchronization rule are discarded.

The category $\lambda\overline{Nets}$ has the collection of labeled nets as objects, and label preserving morphisms as arrows. As a restriction of $\times_{\overline{N}}$, it is easy to check that the product defined above is the categorical product $\times_{\lambda\overline{N}}$ in $\lambda\overline{Nets}$. The categories $\lambda\overline{Occ}$ and $\lambda\overline{Tr}$ can be derived from $\lambda\overline{Nets}$ exactly as before, and have the same relations. In particular, the product $\times_{\lambda\overline{T}}$ (resp. $\times_{\lambda\overline{O}}$) of labeled trellis nets (resp. occurrence nets) can be obtained by 1/ taking the standard product of non-labeled trellis nets (resp. occurrence nets), and 2/ removing transitions not matching the rules of the synchronization algebra. A recursive definition of these products is given in appendix C.

5.2 Factorization in elementary trellises

With this simple synchronization algebra, every multi-clock net $\mathcal{N} = (P, T, \rightarrow, P^0, \mu)$ can be viewed as a synchronous product of sequential machines (fig. 12). Add to $\mathcal{N}$ the label set $\Lambda = T$ and take the identity as labeling function λ. Then consider each restriction $\mathcal{N}_{|\bar{p}}$ for $p \in P^0$, with $T_{|\bar{p}} = \{t \in T : {}^\bullet t \cap \bar{p} \neq \emptyset\}$ as transition set, and $\Lambda = T_{|\bar{p}}$ as label set. By definition of multi-clock nets, $\mathcal{N}_{|\bar{p}}$ is a sequential machine, and clearly $\mathcal{N} = \times_{\lambda\overline{N},\, p \in P^0} \mathcal{N}_{|\bar{p}}$. This decomposition justifies *a posteriori* the name "multi-clock net:" each automaton $\mathcal{N}_{|\bar{p}}$ has a natural notion of time, that we use to compute the height function.

Applying (11) to such a decomposition reveals that the time-unfolding of a net $\mathcal{N}$ is the product (in $\lambda\overline{Tr}$) of the time unfoldings of its components $\mathcal{N}_{|\bar{p}}$. And the latter are nothing more than ordinary automata trellises, as they are usually understood by several communities (fig. 13). This nice property adds substance to the claim that trellis nets are the correct generalization to concurrent systems of the ordinary notion of trellis.

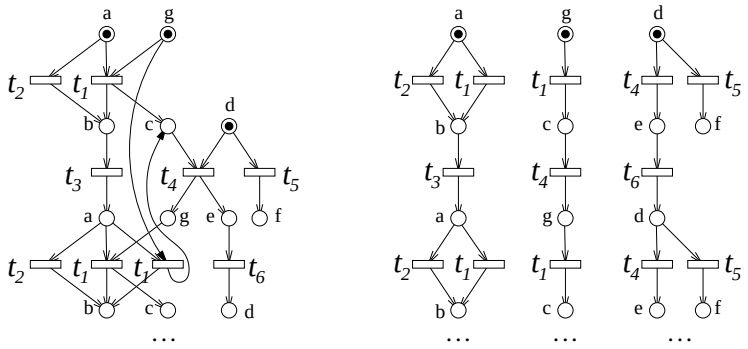

Figure 13: *Left: Trellis of the net $\mathcal{N}$ depicted on fig. 12. Right: Trellises of its components $\mathcal{N}_{|\bar{a}}, \mathcal{N}_{|\bar{g}}, \mathcal{N}_{|\bar{d}}$. The trellis on the LHS is the product of the three other trellises, in the sense of $\lambda\overline{Tr}$: $\mathcal{U}^t_{\mathcal{N}} = \mathcal{U}^t_{\mathcal{N}_{|\bar{a}}} \times_{\lambda\overline{T}} \mathcal{U}^t_{\mathcal{N}_{|\bar{g}}} \times_{\lambda\overline{T}} \mathcal{U}^t_{\mathcal{N}_{|\bar{d}}}$.*

5.3 Other properties of trellises

We take advantage of the previous discussion to illustrate some properties of trellises. By construction, the trellis of a net remains bounded on conditions: the number of conditions having the same height is bounded by a constant. However, this does not hold on events, as one can see in fig. 12: the number of events labeled t_1 in $\mathcal{U}^t_{\mathcal{N}}$ increases with height. This phenomenon is in favor of factorized forms of trellises: although it keeps the number of conditions under control, the product of trellises augments the number of events, as in the case of unfoldings (see for example fig. 14).

We have mentioned in section 2.4 that the restriction to multi-clock nets was harmless: one can always add complementary places to a safe net and make it multi-clock, without changing its behaviour. This operation has little impact on the construction of unfoldings: let $\mathcal{N}$ be a safe net and $\bar{\mathcal{N}}$ its complemented version, then $\mathcal{U}_{\mathcal{N}}$ can be recovered by erasing conditions pointing to complementary places in $\mathcal{U}_{\bar{\mathcal{N}}}$. Things are different with trellises, as shown by the example in fig. 15. This net has a single sequential component. By adding complementary places, one artificially creates three sequential components, and thus three clocks instead of one. Although configurations of $\mathcal{N}$ and $\bar{\mathcal{N}}$ are in a simple one to one correspondence, trellis shapes are strongly different, due to different ways of computing heights.

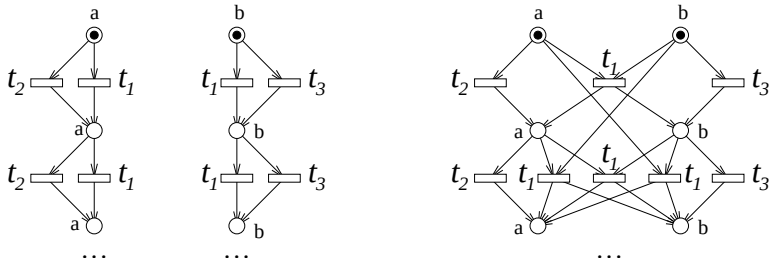

Figure 14: *Two labeled TPs (left), sharing label t_1, and their product (right). This example shows that the factorization of TP has little impact on the number of conditions, but significantly reduces the number of events, and the complexity of the TP. The factors on the left have constant width at a given height, but the product explodes.*

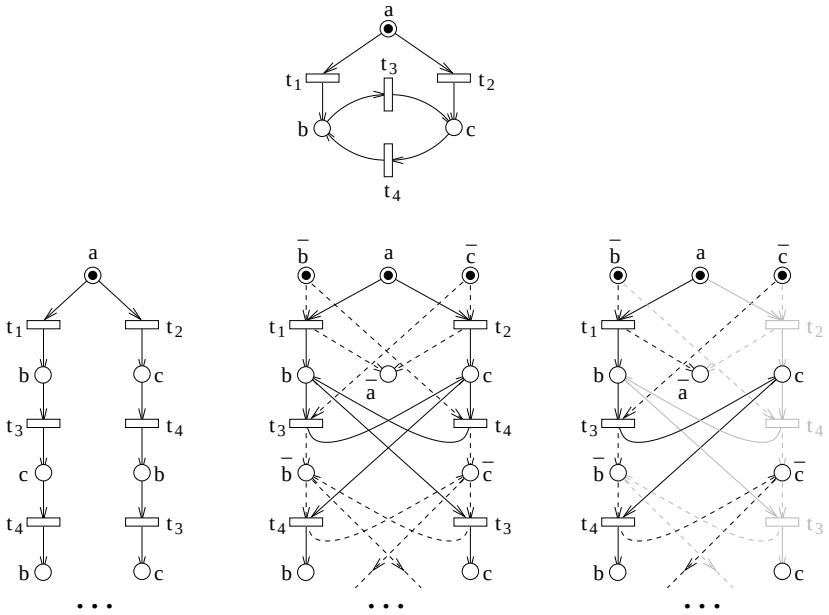

Figure 15: *A net $\mathcal{N}$ (top), its trellis (bottom left), and the trellis of its complemented version $\bar{\mathcal{N}}$ (center). Dashed lines underline the effect of complementary places. One of the two infinite configurations of $\mathcal{U}^t_{\bar{\mathcal{N}}}$ is enlighted in the rightmost net.*

6 Variations around the height function

Refering to (16), trellis processes are obtained as a maximal (conflict) folding of a branching process, guided by a height function H. To ensure all properties of the previous sections, the latter must essentially satisfy two conditions :

1. H must be a causal function, *i.e.* must depend only on the past of a condition in a given string, in order to allow recursive constructions of trellis processes ;
2. and H must be a monotonic function, in order to prevent the creation of circuits in each $\mathcal{T}_{|\bar{c}}$.

This leaves a fair amount of flexibility that we explore now.

6.1 A causal monotonic function

Let $(\mathcal{E}, <)$ be a partially ordered set. A *height function* H taking values in $\mathcal{E}$ is a collection of functions H_σ, for $\sigma = (C_\sigma, E_\sigma, \rightarrow, C^0_\sigma, \nu)$ a string, with $H_\sigma : C_\sigma \rightarrow \mathcal{E}$, and such that H is invariant by string isomorphism. H is said to be *causal* iff it satisfies

$$\forall \sigma, \ \ \forall c \in C_\sigma, \quad H_\sigma(c) \ = \ H_{[c]}(c) \tag{21}$$

where $[c]$ denotes the minimal sub-string of σ containing c. H is *monotonic* iff

$$\forall \sigma, \ \ \forall c, c' \in C_\sigma, \quad c \rightarrow^* c' \ \ \Rightarrow \ \ H_\sigma(c) < H_\sigma(c') \tag{22}$$

The definition of a causal monotonic H given above introduces little flexibility since two strings are isomorphic as soon as they have the same length. So we basically rephrased (6)... Things change when one considers richer structures. For instance, let us define *weighted nets* (associated to weight-preserving morphisms) as $\mathcal{N} = (P, T, \rightarrow, P^0, \nu, w)$, with $w : P \cup T \rightarrow \mathbb{R}^+$. On a weighted string σ, one can define

$$H_\sigma(c) \ = \ \sum_{e \in E_\sigma, \ e \rightarrow^* c} w(e) \tag{23}$$

$$\text{or} \qquad H_\sigma(c) \ = \ \sum_{x \in C_\sigma \cup E_\sigma, \ x \rightarrow^* c} w(x) \tag{24}$$

As another example, let us consider labeled nets $\mathcal{N} = (P, T, \rightarrow, P^0, \nu, \lambda, \Lambda)$, where all label sets Λ are included in $\mathcal{L}$. Take $\mathcal{E} = \mathcal{L}^*$, and let $<$ be the prefix relation in $\mathcal{L}^*$. In a labeled string $\sigma = (C_\sigma, E_\sigma, \rightarrow, C^0_\sigma, \nu, \lambda, \Lambda)$ corresponding to the sequence $c_0 \rightarrow e_1 \rightarrow c_1 \rightarrow e_2 \rightarrow \ldots \rightarrow e_n \rightarrow c_n \, (\rightarrow \ldots)$ one can define

$$H_\sigma(c_n) \ = \ \lambda(e_1)|\lambda(e_2)|\ldots|\lambda(e_n) \tag{25}$$

where $|$ denotes the concatenation operator.

Lemma 4 *Given the height function H defined by (25), consider the labeled net $\mathcal{N} = (P, T, \rightarrow, P^0, \nu, \lambda, \Lambda)$. If λ is injective, the trellis $\mathcal{U}^t_\mathcal{N}$ is isomorphic to the unfolding $\mathcal{U}_\mathcal{N}$.*

Proof. Let $(\mathcal{O}, \phi)$ be a branching process of $\mathcal{N}$, and consider its restriction $\mathcal{O}_{|\bar{c}}$ for $c \in C^0$. If λ is injective in $\mathcal{N}$, it is also injective in any of its sequential components, and in particular in $\mathcal{N}_{|\bar{p}}$ for $p = \phi(c) \in P^0$. With this remark, two conditions of $\mathcal{O}_{|\bar{c}}$ have the same height iff they have been generated by the same sequence of transitions of $\mathcal{N}$, and thus are identical. Therefore $\mathcal{O}$ is correctly and maximally folded for H, which defines a trellis process of $\mathcal{N}$. □

It is straightforward to check that, given a causal monotonic height function, correctly folded trellis nets enjoy the same properties as before. Time unfoldings can still be defined and satisfy theorem 2.

6.2 Different merge rules in the sequential components

A height function operates on runs of a sequential component, but with the definitions above, H is the same for all sequential components of all nets. We describe here a mechanism that allows a fine tuning of the height function, according to the component to which it applies. We start by attaching a "type" to each sequential component of a MCN. Let $\mathcal{A}$ be a set of possible types, a *typed net* $\mathcal{N} = (P, T, \rightarrow, P^0, \nu, \tau)$ is a MCN enriched with a function $\tau : P^0 \rightarrow \mathcal{A}$. The type function τ extends to all places $p \in P$ by $\tau(p) \triangleq \tau(\nu(p))$. We naturally limit ourselves to type preserving morphisms.

It is now possible to define different height functions, according to the component type: a *typed height function* is a collection of height functions $H_{\cdot,\alpha}$ operating on strings of type $\alpha \in \mathcal{A}$. The height of condition c in a typed configuration κ is naturally given by $H_{\cdot,\tau(c)}$. With the extra requirement that each $H_{\cdot,\alpha}$ is causal and monotonic, one recovers all properties of trellis nets.

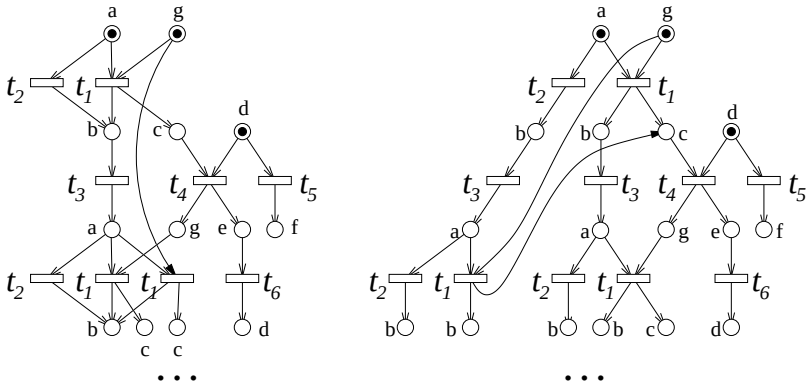

Figure 16: *Two trellises of the net $\mathcal{N}$ of fig. 12. On the left, the height function doesn't refold runs of the sequential component $\mathcal{N}_{|\bar{g}}$. Observe the two occurrences of c on the bottom line. Conversely, runs of $\mathcal{N}_{|\bar{a}}$ are not refolded on the RHS trellis.*

Combining this mechanism with the ideas of section 6.1 allows a fine tuning of the "refolding degree" performed by the trellis of a net. For example, it is possible to decide that in $\mathcal{U}_{\mathcal{N}}$, runs of a given sequential component will remain completely unfolded, whereas for another component, the usual trellis structure will be chosen. Consider the running example of fig. 11, where $\mathcal{N}$ has three sequential components $\mathcal{N}_{|\bar{a}}$, $\mathcal{N}_{|\bar{g}}$ and $\mathcal{N}_{|\bar{d}}$. Let H be chosen as in (6) for all components excepted $\mathcal{N}_{|\bar{g}}$, for which H imposes no merge at all. One gets the LHS trellis in fig. 16. Conversely, if runs of $\mathcal{N}_{|\bar{a}}$ aren't merged, one gets the RHS trellis. Both satisfy the universal property (for the corresponding choice of height function), and thus enjoy the factorization property (11).

In summary, playing with the height function, the central category $D = \overline{Tr}$ in fig. 10 can be shifted horizontally to the left, up to coinciding with $C = \overline{Occ}$.

7 Conclusion

To study properties of concurrent systems, one needs an efficient tool to represent all their possible runs. In the true concurrency semantics, this is generally done by means of branching processes of the system, *i.e.* by prefixes of its unfolding. We have introduced a more compact structure than the unfolding, the trellis, which is a partial refolding of the former, guided by a simple height function. The price to pay for having a more compact structure is a more complex procedure to extract configurations, *i.e.* runs of the system. However, by an appropriate choice of the height function, one has at least a means to adjust this classical tradeoff between memory and computation complexity. The height function is indeed the element that pilots the refolding degree of conflicts.

Like the unfolding, the trellis of a concurrent system enjoys factorization properties : When a system can be expressed as a product of components, its trellis itself is the product (in an appropriate sense) of the trellises of these components. The factorized form of a trellis is of course even more compact, and the factors are much simpler than their product. This suggests a first strategy to simplify the extraction of configurations, by operating "by parts." By the way, let us mention that applications of unfoldings have apparently not taken advantage of this factorization property. To be able to work on factorized forms of a trellis (or of an unfolding), one needs to develop a projection operator, and check that projection and product jointly satisfy a small set of axioms [24]. This will be detailed in a forthcoming paper. In the same way, a standard problem about unfoldings is the derivation of a finite complete prefix, which was the key to make unfolding techniques a practical tool. In this paper, we have chosen to put aside these aspects. But the derivation of a (minimal) complete prefix of $\mathcal{U}^t(\mathcal{N})$, and its relations with complete prefixes of $\mathcal{U}(\mathcal{N})$, remain problems of considerable interest.

Acknowledgement : The author is grateful to Philippe Darondeau and Glynn Winskel for fruitful (and patient) exchanges in the elaboration of early versions of this work.

References

[1] S. Mac Lane, Categories for the Working Mathematician, Springer-Verlag, 1971.

[2] M. Nielsen, G. Plotkin, G. Winskel, Petri nets, event structures and domains, Theoretical Computer Science 13(1), 1981, pp. 85-108.

[3] G. Winskel, A new Definition of Morphism on Petri Nets, LNCS 166, pp. 140-149, 1984.

[4] G. Winskel, Categories of models for concurrency, Seminar on Concurrency, Carnegie-Mellon Univ. (July 1984), LNCS 197, pp. 246-267, 1985.

[5] G. Winskel, Event structure semantics of CCS and related languages , LNCS 140, 1982, also as report PB-159, Aarhus Univ., Denmark, April 1983.

[6] G. Winskel, Petri Nets, Algebras, Morphisms, and Compositionality , Information and Computation, no. 72, pp. 197-238, 1997.

[7] J. Engelfriet, Branching Processes of Petri Nets, Acta Informatica no. 28, pp. 575-591, 1991.

[8] F. W. Vaandrager, A simple definition for parallel composition of prime events structures, Report CS-R8903, CWI, Amsterdam, March 1989.

[9] K.L. McMillan, Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits, in Proc. 4th Workshop of Computer Aided Verification, Montreal, 1992, pp. 164-174.

[10] K.L. McMillan, Symbolic Model Checking: An Approach to the State Explosion Problem, PhD thesis, Kluwer, 1993.

[11] J. Esparza, Model checking using net unfoldings, Science of Computer Programming 23, pp. 151-195, 1994.

[12] J. Esparza, S. Römer, W. Vogler, An improvement of McMillan's unfolding algorithm, in Proc. of TACAS'96, LNCS 1055, pp. 87-106.

[13] J. Esparza, S. Römer, W. Vogler, An Improvement of McMillan's Unfolding Algorithm, Formal Methods in System Design 20(3), pp. 285-310, May 2002. Extended version of [12].

[14] J. Esparza, S. Römer, An unfolding algorithm for synchronous products of transition systems, in Proc. of CONCUR'99, LNCS 1664, Springer Verlag, 1999.

[15] J. Esparza, C. Schröter, Reachability Analysis Using Net Unfoldings, Workshop of Concurrency, Specification and Programming, volume II of Informatik-Bericht 140, pp. 255-270, Humboldt-Universität zu Berlin, 2000.

[16] S. Melzer, S. Römer, Deadlock checking using net unfoldings, CAV'97, LNCS 1254, pp. 352-363.

[17] J.-M. Couvreur, S. Grivet, D. Poitrenaud, Unfolding of Products of Symmetrical Petri Nets, 22nd International Conference on Applications and Theory of Petri Nets (ICATPN 2001), Newcastle upon Tyne, UK, June 2001, LNCS 2075, pp. 121-143.

[18] V. Khomenko, M. Koutny, W. Vogler, Canonical Prefixes of Petri Net Unfoldings, Acta Informatica, vol. 40, pp. 95-118, 2003

[19] V. Khomenko, A. Kondratyev, M. Koutny, W. Vogler, Merged Processes - a New Condensed Representation of Petri Net Behavior, Tech. Rep. Series CS-TR-884, Univ. of Newcastle upon Tyne, Jan. 2005.

[20] A. Benveniste, E. Fabre, S. Haar, C. Jard, Diagnosis of asynchronous discrete event systems, a net unfolding approach, IEEE Trans. on Automatic Control, vol. 48, no. 5, pp. 714-727, May 2003.

[21] E. Fabre, Factorization of Unfoldings for Distributed Tile Systems, Part 1 : Limited Interaction Case, Inria research report no. 4829, April 2003.

[22] E. Fabre, Factorization of Unfoldings for Distributed Tile Systems, Part 2 : General Case, Inria research report no. 5186, May 2004.

[23] E. Fabre, A. Benveniste, S. Haar, C. Jard, Distributed Monitoring of Concurrent and Asynchronous Systems, Journal of Discrete Event Systems, special issue, to appear in March 2005.

[24] E. Fabre, Convergence of the turbo algorithm for systems defined by local constraints, Irisa research report no. PI 1510, May 2003.

[25] C. Cassandras, S. Lafortune, Introduction to Discrete Event Systems, Kluwer Academic Publishers, 1999.

[26] W. Reisig, Petri Nets, Springer Verlag, 1985.

A Some keypoints about adjunctions

This section briefly recalls some elements of category theory. Its objective is to show how a few key results trigger standard constructions, from which we derive most properties mentioned without proof in the paper. The paper will thus concentrate on these key results. We assume the reader is familiar with the basic notions of category, functor and natural transformation of a functor into another [1].

A.1 Product

Let o_1 and o_2 be two objects in a category C, and consider triples (o, f_1, f_2) where o is an object in C, and the $f_i : o \to o_i$ are morphisms, $i = 1, 2$. The *product* of o_1 and o_2 in C, if it exists, is defined as such a triple $o_1 \times_C o_2 \triangleq (p, \psi_1, \psi_2)$ which is also required to be extremal. Specifically,

$$\forall (o, f_1, f_2),\ \exists! \phi : o \to p, \quad [f_1 = \psi_1 \circ \phi \ \text{ and } \ f_2 = \psi_2 \circ \phi] \tag{26}$$

This condition is referred to as the universal property of the product (fig. 17). The product, when it exists, is unique up to isomorphism.

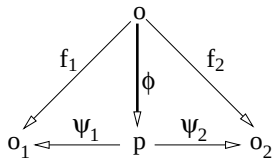

Figure 17: *Universal property of the product, expressed as a commutative diagram.*

A.2 Adjunction

This is probably the central notion of this paper: it forms the basis of most results we state. An *adjunction* between two categories C and D is defined as a triple (F, G, ϕ). $F : C \to D$ and $G : D \to C$ are two functors relating C and D, and working in opposite directions. F and G are respectively the left and right adjoints. ϕ is a bijective correspondence between morphisms of the two categories. Specifically, for any two objects $c \in C$ and $d \in D$, $\phi_{c,d} : \mathrm{Mor}_D(F(c), d) \to \mathrm{Mor}_C(c, G(d))$ is bijective[11], where $\mathrm{Mor}_X(u, v)$ represents the set of morphisms from u to v in category X. To help intuition, one can view functors as a way of reshaping the objects of one category to match the structure of objects in another category. In an adjunction, one has two opposite ways of doing this reshaping, by F or by G, and Φ explains how relations inside one category are mapped into relations of the other (see fig.18).

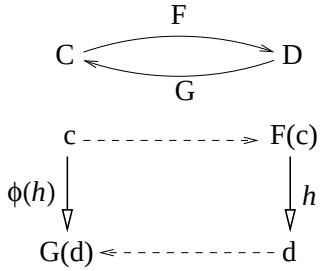

Figure 18: *An adjunction between two categories.*

The mapping ϕ is also required to be "natural in c and in d," which means the following. Let $f : c' \to c$ and $g : d \to d'$ be two morphisms in C and D respectively, and $h \in \mathrm{Mor}_D(F(c), d)$, then

[11] Indexes c, d are often omitted in $\phi_{c,d}$.

(see fig. 19)

$$\phi(g \circ h \circ F(f)) = G(g) \circ \phi(h) \circ f \tag{27}$$

This property can obviously be checked separately in f and in g.

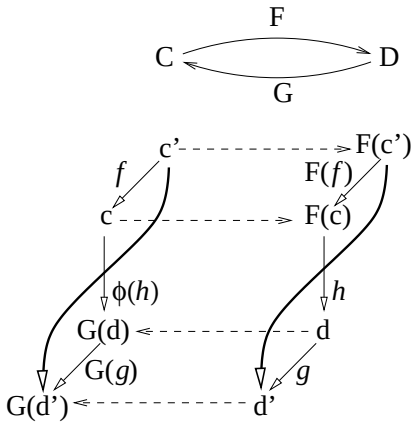

Figure 19: *Naturality of the mapping ϕ in an adjunction.*

A.3 Construction

In this paper, functor F is generally given, and G is defined on objects of C. So the construction of an adjunction amounts to proving that G is indeed a functor (*i.e.* is also defined on morphisms), and to obtaining the correspondence ϕ. The keystone of the construction is the following assumption :

(UA) for every object $d \in D$, there exists a morphism $\epsilon_d : FG(d) \to d$ such that the pair $(G(d), \epsilon_d)$ forms a universal morphism from (images of) functor F to d :

$$\forall c \in C,\ \forall g : F(c) \to d,\ \exists! f : c \to G(d) \quad g = \epsilon_d \circ F(f) \tag{28}$$

In words, all morphisms from $F(c)$ to d factorize through $FG(d)$.

From this assumption, the adjunction is derived in the following way ([1], chap. IV-1, thm 2-iv).

One first proves that G can be extended to morphisms of D, and form a functor $G : D \to C$. Let $g : d \to d'$ be an arrow of D, and consider morphism $g \circ \epsilon_d : FG(d) \to d'$. By (UA), there exists a unique $f : G(d) \to G(d')$ in C such that $\epsilon_{d'} \circ F(f) = g \circ \epsilon_d$. We define $G(g) \triangleq f$, so

$$g \circ \epsilon_d \quad = \quad \epsilon_{d'} \circ FG(g) \tag{29}$$

It is then easy to show that G is a functor : $G(1) = 1$ and $G(g_2 \circ g_1) = G(g_2) \circ G(g_1)$.

Property (29) expresses that $\epsilon : FG \to \mathbb{I}_D$ is a natural transformation of functor FG to the identity functor $\mathbb{I}_D$ in D. The correspondence ϕ is then derived in the following way :

$$\forall h : c \to G(d), \qquad \phi^{-1}(h) \quad \triangleq \quad \epsilon_d \circ F(h) \tag{30}$$

which implies in particular $\epsilon_d = \phi^{-1}(1_{G(d)})$, see fig. 20. By (UA), one checks that $(\phi^{-1})^{-1}$ is well defined, i.e. that ϕ is bijective. By definition of ϕ^{-1}, one has $\phi^{-1}(h \circ f) = \phi^{-1}(h) \circ F(f)$ which proves the naturality of ϕ in c. The naturality of ϕ in d means $\phi^{-1}(G(g) \circ h) = g \circ \phi^{-1}(h)$, which is a direct consequence of (29), i.e. the commutative square on the right-hand side of fig. 20.

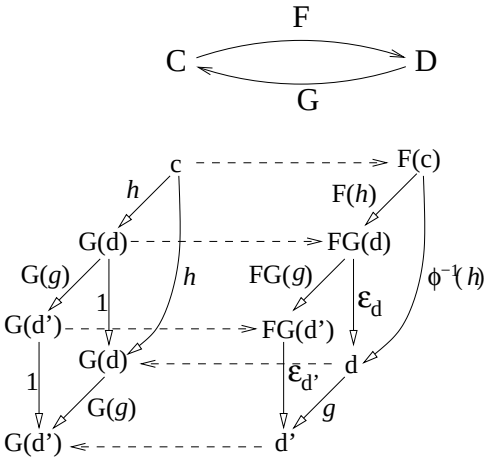

Figure 20: *Derivation of an adjunction, by extending G into a functor, and deriving ϕ from the co-unit ϵ.*

A.4 Properties

Preservation of limits. This is the main result we use on adjunctions: a functor which is a right adjoint preserves limits, and in particular products, which are a special kind of limits ([1], chap. V-5, thm 1). Specifically, let d_1, d_2 be objects in D and let $d_1 \times_D d_2 = (d, \psi_1, \psi_2)$, then $(G(d), G(\psi_1), G(\psi_2))$ satisfy the universal property of the product for $G(d_1)$ and $G(d_2)$ in C, which allows to define $\times_C$ on the objects of C lying in the image of G.

Unit. The natural transformation $\epsilon : FG \to \mathbb{I}_D$ is called the *co-unit* of the adjunction. By symmetry, the *unit* η can be defined as the natural transformation $\eta : \mathbb{I}_D \to GF$ in category C (fig. 21), and ϕ can be recovered from η if the counterpart of (UA) is satisfied.

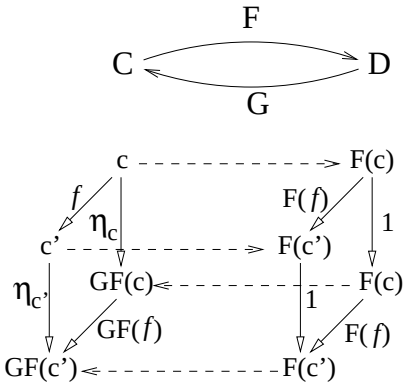

Figure 21: *Unit of an adjunction, from which ϕ can also be reconstructed.*

A special case deserves some interest. Assume

(NE) The unit η is a natural equivalence[12], i.e. has an inverse η^{-1} (which is then also a natural transformation).

Then every object $c \in C$ is isomorphic to $GF(c)$. So if a product $\times_D$ exists in D, this product can be mapped by G into a product on objects of $G(D)$ in C, and finally into a product for all pairs of

[12]The synonym "natural isomorphism" is also frequent.

objects in C, since the product is defined up to isomorphism. One thus has

$$\forall c_1, c_2 \in C, \;\; c_1 \times_C c_2 \cong GF(c_1) \times_C GF(c_2) = G(F(c_1) \times_D F(c_2)) \tag{31}$$

Composition of adjunctions. When (F, G, ϕ) is an adjunction between C and D, and (F', G', ϕ') an adjunction between D and E, one easily checks that $(\bar{F}, \bar{G}, \bar{\phi}) \triangleq (F' \circ F, G \circ G', \phi \circ \phi')$ defines an adjunction between C and E ([1], chap. IV-8, thm 1). Its co-unit $\bar{\epsilon}$ is given by

$$\forall c \in C, \;\; \bar{\epsilon}_c = G(\epsilon'_{F(c)}) \circ \epsilon_c \tag{32}$$

and its unit $\bar{\eta}$ is given by

$$\forall e \in E, \;\; \bar{\eta}_e = \eta'_e \circ F'(\eta_{G'(e)}) \tag{33}$$

B On the choice of net morphisms

This section illustrates the fact that, if one does not allow morphisms that duplicate places of a net, the resulting category *Nets* doesn't have a categorical product, and so is necessarily incomplete[13]. We provide a simple counter-example: two nets for which the standard construction of product violates the universal property.

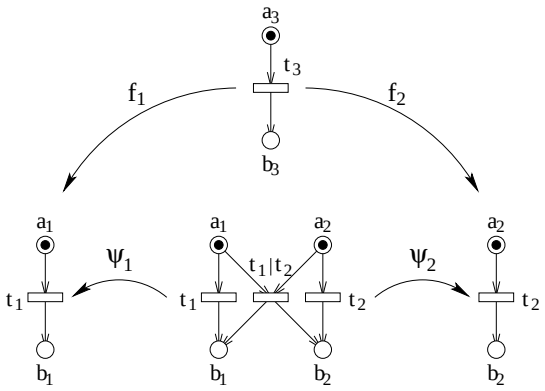

Figure 22: *The universal property of the product requires morphisms with the ability to duplicate places.*

Consider the three isomorphic nets $\mathcal{N}_i$, $i \in \{1, 2, 3\}$, each composed of a single transition t_i, with one input place a_i (initially marked) and one output place b_i (empty). By construction, there exist unique morphisms ψ_i from (what should be) the product $\mathcal{N}_1 \times_N \mathcal{N}_2$ to factors $\mathcal{N}_i$, $i \in \{1, 2\}$ (see fig. 22). In the same way, there exist isomorphisms $f_i : \mathcal{N}_3 \to \mathcal{N}_i$. Finally, the unique morphism $\phi : \mathcal{N}_3 \to \mathcal{N}_1 \times_N \mathcal{N}_2$ that makes the diagram commutative, *i.e.* that satisfies $f_i = \psi_i \circ \phi$, is such that it duplicates places of $\mathcal{N}_3$: a_3 is mapped to a_1 and a_2 in the product, b_3 to b_1 and b_2, and t_3 is mapped to the synchronized transition (t_1, t_2). So if morphisms duplicating places are not allowed, the commutative diagram cannot be constructed, and the universal property of the candidate product $\mathcal{N}_1 \times_N \mathcal{N}_2$ is lost.

C Product of labeled trellis nets

Let $\mathcal{T}_i = (C_i, E_i, \to_i, C_i^0, \nu_i, \lambda_i, \Lambda_i)$, $i = 1, 2$, be two labeled trellis nets. Their product $\mathcal{T} = (C, E, \to, C^0, \nu, \lambda, \Lambda = \Lambda_1 \cup \Lambda_2)$ is defined by $\mathcal{T} = \mathcal{T}_1 \times_{\lambda\overline{T}} \mathcal{T}_2 \cong \mathcal{U}^t(\mathcal{T}_1 \times_{\lambda\overline{N}} \mathcal{T}_2)$. By merging the definition of

[13]Notice that, even with morphisms able to duplicate places, the category *Nets* remains incomplete, for a different reason that we do not detail here. But at least *Nets* has a product.

$\times_{\lambda\overline{N}}$ with the recursive construction of $\mathcal{U}^t(\mathcal{N})$ for a net $\mathcal{N}$, one gets the following recursive form for the product $\times_{\lambda\overline{T}}$:

Procedure 7

- Initialization :
  - Create $|C_1^0| + |C_2^0|$ conditions in C^0, and define injective partial functions $\psi_i : C^0 \to C_i^0$ in such a way that they have disjoint domains.
  - Set $C = C^0$, $E = \emptyset$, $\rightarrow = \emptyset$, $\nu = Id$ and $\Lambda = \Lambda_1 \cup \Lambda_2$.
- Recursion :
  - Let X be a co-set of C, $\alpha \in \Lambda$ and $I = \{i : \alpha \in \Lambda_i\}$;
    $\forall i \in I$, let e_i be an event of E_i such that $\lambda_i(e_i) = \alpha$ and ${}^\bullet e_i = \psi_i(X)$.
  - If there doesn't exist an event $e \in E$ with ${}^\bullet e = X$, $\lambda(e) = \alpha$ and $\forall i \in I,\ \psi_i(e) = e_i$,
    - create a new event $e \in E$ with ${}^\bullet e = X, \lambda(e) = \alpha$ and $\forall i \in I$ set $\psi_i(e) = e_i$,
    - create a subset Y of $\sum_{i\in I} |e_i^\bullet| = |X|$ new conditions in C, set $e^\bullet = Y$,
    - extend the partial functions $\psi_i,\ i \in I$, to Y in order to have disjoint definition domains in Y and to satisfy $\psi_i : Y \to e_i^\bullet$ injective[14],
    - define ν on Y by $\nu(c) = \psi_i^{-1} \circ \nu_i \circ \psi_i(c)$ for the unique $i \in I$ where $\psi_i(c)$ is defined,
    - $\forall i \in I,\ \forall c \in Y$, if $\exists c' \in C,\ \psi_i(c') = \psi_i(c)$ and $H(c') = H(c)$ then merge conditions c and c'.

The choice of creating new conditions that may disappear afterwards in a merge operation is somehow inelegant. However, this formulation has a nice advantage: procedure 7 without the final merge operation actually computes the product of labeled *occurrence nets*.

D Comparison with "merged processes"

Some authors have proposed the notion of *merged processes* [19], which shares many similarities with trellis processes. In the same way, the time unfolding of a net would be the *unravelling* of [19]. We briefly relate these notions here, and stress some differences.

Merged processes (MP) are obtained by a partial refolding of the branching processes of a net (see the comment after eq. (16)). They are defined for general nets, provided places contain at most one token in the initial marking. This broad scope apparently prevents a categorical treatment of their construction. The refolding criterion is based on a height function that is called the *occurrence-depth* in [19]. Specifically, let us limit ourselves to safe nets, for a comparison. Let $\mathcal{N} = (P, T, \rightarrow, P^0)$ be a general safe net, and consider the safe net $\bar{\mathcal{N}} = (\bar{P}, \bar{T}, \rightarrow', \bar{P^0}, \nu)$ obtained by adding a complementary place to each place of $\mathcal{N}$ ($\bar{\mathcal{N}}$ and $\mathcal{N}$ have the same behavior). $\bar{\mathcal{N}}$ becomes a multiclock net when each place is associated to its complementary place, so the n places of P give rise to n sequential components in $\bar{\mathcal{N}}$. By computing the time-unfolding of $\bar{\mathcal{N}}$, then erasing all conditions of $\mathcal{U}^t(\bar{\mathcal{N}})$ pointing to complementary places, one obtains the unravelling of $\mathcal{N}$.

Despite this very tight link between the two notions, the last operation (the removal of complementary conditions) has a dramatic effect on the resulting structure because executable cycles are introduced. Consider for example the sequential machine $\mathcal{N}$ of fig. 23 (left). The trellis of $\bar{\mathcal{N}}$ (fig. 23, center) contains cycles, but the latter can't be executed, precisely because of the presence of complementary places. So $\mathcal{U}^t(\bar{\mathcal{N}})$ essentially contains two infinite configurations (see one of them in fig. 15).

[14] on its domain of definition

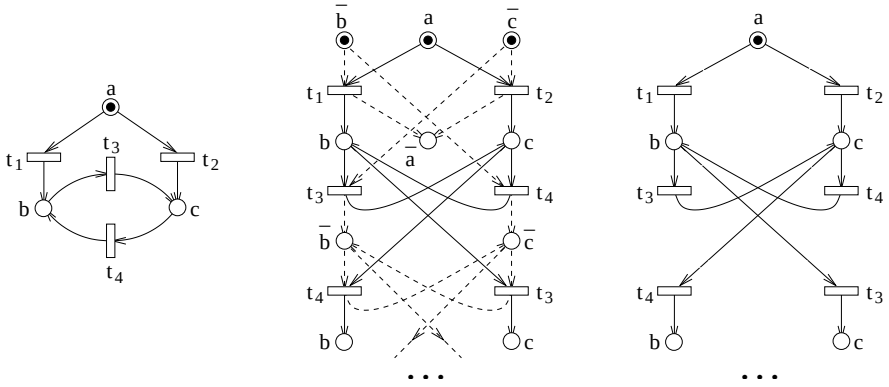

Figure 23: *A sequential machine $\mathcal{N}$ (left). The trellis of $\bar{\mathcal{N}}$,* i.e. *$\mathcal{N}$ enriched with complementary places (center). Dashed lines are associated to the treatment of complementary conditions. The removal of complementary conditions yields the "unravelling" of $\mathcal{N}$ (right), which contains executable cycles. For example the circuit t_3, t_4 on this prefix.*

However, when complementary conditions are removed, it becomes possible to execute these cycles (fig. 23, right), whence the problem of "spurious markings" in [19].

Properties of MPs given in [19] are essentially obtained as properties of the corresponding BP, before the folding operation. The fact that all runs of the unravelling of $\mathcal{N}$ can be mapped to runs of $\mathcal{N}$ remains to be proved. However, experimental results already show that by folding a complete prefix of $\mathcal{U}_{\mathcal{N}}$, one gets a much more compact structure, on which model-checking problems can be solved more efficiently.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399