


Università degli Studi di Cagliari

Dipartimento di Matematica e Informatica
Ph.D. Course in Computer Science
Cycle XXVIII

Ph.D. Thesis

Unravel nets

A way to represent behaviors compactly and to relate them to event structures

S.S.D. INF/01

Candidate:
Giovanni Casu

Ph.D. Coordinator:
Giovanni Michele Pinna

Supervisor:
Giovanni Michele Pinna

Academic year 2015/2016

"A witty saying proves nothing."
Voltaire

Abstract

The present work introduces a new class of nets which aims to give a more compact non-sequential semantics to safe Petri nets, namely *unravel nets*. As causal nets can be a representation of runs of a safe Petri net, *i.e.* its unfolding, unravel nets can be regarded as a succinct version of these unfoldings. The main contributions of the thesis, beside the definition of this class of nets, are:

- their close connection with a brand of event structures, namely *bundle* event structures, and we show that configurations of the former can be mapped into the ones of the latter and vice versa,
- the encodings, in terms of unravel nets, of the existing approaches for merging unfoldings,
- the definition of a general notion of *merging relation* which can, under certain constraints, preserve some properties of a net, in particular we introduce the notion of *conflict conditions* which can force a net to be an unravel one after the merging,
- the addition of *contextual* arcs (in our approach *read arcs*) to unravel nets, which allow to model new kinds of causality. We consider various kind of event structure with *non-standard* causality, namely *dynamic causality* event structure, and prove that they are related to contextual unravel nets similarly to what happens to bundle event structures and unravel nets without contexts.

Contents

List of Figures

Chapter 1

Introduction

1.1 Scenario

Petri nets [Pet66, Rei85, Rei13] are a well-known model for describing distributed systems.

In order to represent how a system evolves, *i.e.* to model the behavior of the system, the semantics are essentially of two kind: *sequential* and *non-sequential*. The former takes into account how different markings changes from the initial one, and are based on the analysis of the so called *reachability graph*. Each node represents a marking and there is a directed arc from one node to another if from that marking it is possible to fire a transition which marks the correspondent places of the second node. The reachability graph is often related to the *state space explosion* problem: graph size is, in general, exponentially larger than the original net. This growth in size can be attributed to the concurrency that can can arise among transitions. Also each execution assumes a total order among transitions fired.

Non-sequential semantics instead models causal dependencies among transitions by casting their possible executions onto partial order of events (concurrency means absence of order). This allows in part to reduce the state space explosion:

runs of the system do not take into account all possible interleaves of transitions, which can be deduced directly from the relations of conflict and dependency, and concurrency is handled by allowing the execution of events in any order.

Although non-sequential semantics give a larger structure than the original net, they have in general a better performance. Another aspect that makes them attractive is that partial order are strongly tied to a specific class of Petri nets, the *causal* ones. Causal nets are acyclic safe nets (for each execution at most one token can be found in a place), where places have at most an incoming arc and where the events can be partially ordered.

Thus, it is possible, given a Petri net, to associate a causal net, with a suitable labeling for events and conditions, in which each run is mapped into a run of the starting net. Such causal nets are called unfoldings. From the fact that an unfolding can be naturally equipped with a partial order (among its events) it follows a fundamental result in the field of Petri net semantics: the connection between causal nets and *prime event structures* [NPW81, Win87, Win88].

Indeed, the relations of dependency and conflict among events in a causal net can be easily defined and they satisfy the requirements for a prime event structure.

The use of unfoldings for net semantics in practical applications needs to face with the infinite or too large size that they can often have. Techniques exists to generate a finite representation of an unfolding which contains enough *information*. Nonetheless it may happen that this finite net can be still exponential with respect to the size of the original net.

To cope with this problems two known methods exist which condense an unfolding by *merging* places and transitions: *trellis processes* [Fab07] and *merged processes* [KKKV06]. Driven by different motivations and with similar but substantially different results they achieve the goal by finding equivalences among places and transitions. Those equivalences are found by considering as equal suffixes of conflicting computations that satisfy certain properties, which is, roughly

speaking, the fact that these suffixes are originated at the same *state*.

Although such techniques have proven their usefulness into practice, the resulting nets loses the property of being causal,*i.e.* cycles (syntactic or semantic) can arise after the merging, thus removing any connection with prime event structures. Moreover it is not clear if other kind of event structures can be related to them.

1.2 Contribution and outline of the thesis

This work is centered on the notion of *unravel net*. Unravel nets are defined as safe Petri nets where each execution, intended as the set of transitions that can be fired starting from the initial marking, identifies a causal net. Unravel nets try to capture the idea of acyclic runs of a net whereas the unravel nets themselves, are not, in general, acyclic. The aim is to use them to represent compact behaviors of Petri nets on the one hand, and to relate them to event structures on the other hand. Many authors have considered suitable classes of nets and have related them to suitable notions of event structures. We may recall, among others, the *flow nets* and flow event structures[Bou90] , *contextual nets*[MR95] and asymmetric event structures, *inhibitor nets* and inhibitor event structure or the *1-occurrence net* and their corresponding notion of *event* and *configuration structures* [vP09]. The event structures brand taken into consideration here as counterpart of unravel nets are the *bundle event structures* [Lan93]. Due to the syntactical properties of the unravel nets, it is easy to associate a bundle event structure (BES). They allows to model structural features of the nets such has cycles (not executable) which are not allowed in causal nets. Furthermore there is a one to one correspondence between traces of an unravel nets and those of bundle event structures. In [CP14] we introduced an early characterization of unravel nets which we tied to the *flow* event structures. Unfortunately we faced some issues when extending the notion to any kind of safe nets different from multi-clock nets.

In addition, we show how from a BES an unravel net can be built, maintaining

the correspondence between traces.

We observe that existing techniques for compacting nets do not build, in general, an unravel net. The problem arises because merging histories leads to forgetting past conflicts, and this allows places to be marked again in certain computations, *i.e.* syntactic cycles can be traversed, which is not allowed in unravel nets. Given that, we introduce a way to force such nets to be unravel. We introduce the idea of adding suitable places that have the purpose of maintaining lost conflicts among transitions. We show that although trellises are unravel nets by themselves, merged process can indeed be turned into them and that there is a bijection between the configurations, *i.e.* enriching those nets does not change their behaviors.

Additionally we give a generalization of the merging operation on nets. Both techniques use a criterion for making places equivalent based on their mutual incompatibility. We extend this notion by defining the *merging relations*. With those relations we present a general framework for compacting nets: given an *labeled* unravel net, where the label represent an existing total net morphism, it is possible to compact it giving a merging relation. We distinguish those relations by pointing out when they can lead to unravel nets, thus preserving the *unravelness* property of the original net, or when they need to be enriched. Moreover we prove that trellises and merged processes can be defined as merging relations. This part is an extension of [CP16] and [CP17a].

The last piece of work is based on [CP17b]. At the beginning of the '90s the idea that a partial order was the unique way to represent dependencies has been somehow abandoned. *Bundle event structures* have been introduced to give semantics to LOTOS ([BB87]) and may model *or*-causality and their generalization *dual event structures* ([Kat96]) gain expressivity dropping the assumption that *or*-causality implies that either one or the other cause happens, but not both. These event structures have a more operational flavour with respect to *prime event struc-*

tures ([Win87]), *flow event structures*, *asymmetric event structures* ([BCM01]) or *inhibitor event structure* ([BBCP04]), as the possibility of adding an event is prescribed by bundles that may be sets of events of any kind. This change in perspective has been driven by the observation that the same event (observable activity producing observable changes) could have totally different histories (pasts) and from these histories it was not possible to find a common pattern (see, for instance, the *possible events* in the *event automata* [PP95] or the notion of events in *causal automata*), and it has been further pursued with the notions of asymmetric and inhibitor event structures. However either these phenomena are due to the presence of *contexts* or *inhibitions*, that may be added or removed by the happening of events, or they are represented in a logical way, like in [Gun92] or [vP09]. Event structure with *dynamic* causality are introduced in [AKPN15a] by stipulating that the happening of certain events, called modifiers, may add or remove causal dependencies for a certain event, so that the phenomena of *shrinking* or *growing* causality may be captured. They compare their new notion of event structure to many other presented in the literature with respect to their expressiveness.

The intuition behind all these approaches has always followed a common pattern: to each transition of the net it should be possible to associate an event in the corresponding event structure and from the net structure the relations among events may be deduced.

Since an unravel net can model a bundle event structure and vice versa, we extend unravel nets to cope with the higher expressivity of dynamic event structures compared to the one of bundle event structures. To do so we consider *contextual* unravel nets, *i.e.* we add read arcs [BBCP04, Bal00].

As we have did with bundle event structures, we prove several propositions and theorems relating contextual unravel nets and dynamic event structures, their traces, and how to build one from another.

1.2.1 Outline

- In Chapter 2 we recall basic definitions and notions related to Petri nets, event structures and unfoldings,
- In Chapter 3 we introduce the unravel nets and discuss their relationships with bundle event structures,
- In Chapter 4 the known methods for compacting nets are casted into unravel nets' world. We introduce *conflict* places and show how they can be used to turn merged process into unravel nets. For trellis processes we discuss their tight relationship with unravel nets,
- Merging relations are defined in Chapter 5. Merged and trellis processes are proved to be the result of proper merging relations. A general notion of enrichment of nets is also presented,
- Chapter 6 is aimed to introduce contextual unravel nets and how they can model dynamic event structures. As we did for BES, we prove how traces are preserved when are built from one another.

Chapter 2

Preliminaries

In this chapter we recall the basic definitions, propositions and theorems that we will use in the rest of our work. With $\mathbb{N}$ we denote the set of natural numbers.

2.1 Multiset

Let X be a set, $|X|$ denotes its cardinality. Let A be a set, a *multiset* of A is a function $n : A \to \mathbb{N}$. The set of multisets of A is denoted by μA. The usual operations on multisets, like multiset union $+$ or multiset difference $-$ are, with overloading of notation, defined as usual, hence $(n + n')(s) = n(s) + n'(s)$, $(n - n')(s) = n(s) - n'(s)$ if $n(s) \geqslant n'(s)$ and $(n - n')(s) = 0$ otherwise. We write $n \subseteq n'$ if $n(a) \leqslant n'(a)$ for all $a \in A$. If $n \in \mu A$, we denote by $[\![n]\!]$ the multiset defined as $[\![n]\!](a) = 1$ if $n(a) > 0$ and $[\![n]\!](a) = 0$ otherwise; and we will use $[\![n]\!]$ also as the denotation of the subset $\{a \in A \mid n(a) \geqslant 1\}$ of A. Finally, when a multiset n of A is a set, *i.e.* $m = [\![n]\!]$, we write $a \in n$ to denote that $n(a) \neq 0$, namely that $a \in [\![n]\!]$, and often confuse the multi set n with $[\![n]\!]$, the empty multiset will be confused with the empty set.

2.2 Petri nets

Definition 2.1. *A* Petri net *is a 4-tuple* $N = \langle S, T, F, \mathsf{m} \rangle$*, where*

- S *is a set of* places *and* T *is a set of* transitions *such that* $S \cap T = \varnothing$*,*
- $F \subseteq (S \times T) \cup (T \times S) \to \mathbb{N}$ *is the* flow *relation, and*
- $\mathsf{m} \in \mu S$ *is called the* initial marking.

Petri nets are depicted as usual: places are circles, transitions are boxes and the flow relation is represented by arcs from x to y whenever $F(x, y)$ is greater than 0.

We will consider always labeled nets, namely nets such that to each place or transition a label is associated.

Definition 2.2. *A* labeled *Petri net is* N *is the pair* (N, l)*, where* $N = \langle S, T, F, \mathsf{m} \rangle$ *is a Petri net and* $l \colon S \cup T \to \Lambda$ *a total mapping such that* $l(T) \cap l(S) = \varnothing$*.*

When the label is not relevant, we assume that the labeling function is the identity, and often we will omit it. Sometimes we will write $\langle S, T, F, \mathsf{m}, l \rangle$ instead of $(\langle S, T, F, \mathsf{m} \rangle, l)$.

The flow relation of a net can be seen as a multiset on $(S \times T) \cup (S \times T)$ with the constraint that $\{x, y\} \subseteq S \ \Rightarrow F(x, y) = 0$ and $\{x, y\} \subseteq T \ \Rightarrow F(x, y) = 0$.

We will consider Petri nets where the flow relation F is constrained to be a set, i.e. $\forall x, y.\ F(x, y) \leqslant 1$.

Preset and postset: Given a net $N = \langle S, T, F, \mathsf{m} \rangle$ and $x \in S \cup T$, we define the following multisets: ${}^\bullet x = F(-, x)$, which we call the *preset* of x, and $x^\bullet = F(x, -)$, which we call the *postset* of x.

A transition $t \in T$ is *enabled* at a marking $m \in \mu S$, denoted with $m\,[t\rangle$, whenever ${}^\bullet t \subseteq m$. A transition t enabled at a marking m can *fire* and its firing

produces the marking $m' = m - {}^\bullet t + t^\bullet$. The firing of a transition t at a marking m is denoted with $m\,[t\rangle\,m'$. We assume that each transition t of a net N is such that ${}^\bullet t \neq \varnothing \neq t^\bullet$, which implies that no transition may fire *spontaneously* and that the firing of a transition may have observable effects.

Firing sequences: Given a marking m, the *firing sequence* (fs) starting at m of the net $N = \langle S, T, F, \mathsf{m}\rangle$, is defined as usually:

a) m is a fs, and

b) if $m\,[t_1\rangle\,m_1\ \cdots\ m_{n-1}\,[t_n\rangle\,m_n$ is a firing sequence and $m_n\,[t\rangle\,m'$ then also $m\,[t_1\rangle\,m_1 \cdots m_{n-1}\,[t_n\rangle\,m_n\,[t\rangle\,m'$ is a fs.

The set of firing sequences of a net N starting at a marking m is denoted with $\mathcal{R}_m^N$ and it is ranged over by σ, and we may omit the index denoting the net when it is clear from the context, and the initial marking of the set of firing sequence when it coincides with the initial marking of the net. Given $\mathsf{fs}\ \sigma = m\,[t_1\rangle\,\sigma'\,[t_n\rangle\,m_n$, with $\mathit{start}(\sigma)$ we denote the marking m, with $\mathit{lead}(\sigma)$ the marking m_n and with $\mathit{tail}(\sigma)$ the $\mathsf{fs}\ \sigma'\,[t_n\rangle\,m_n$.

Given a set of markings M, with $\mathsf{P}(M)$ we denote the set of places that are marked at some marking in M, namely $\{s \in S \mid \exists m \in M.\ m(s) > 0\}$, and given a $\mathsf{fs}\ \sigma$, $\mathsf{M}(\sigma)$ are the markings associated to the $\mathsf{fs}\ \sigma$, where $\mathsf{M}(\sigma)$ is $\mathsf{M}(\sigma) = \{m\}$ if $\sigma = m$ and $\mathsf{M}(\sigma) = \{\mathit{start}(\sigma)\} \cup \mathsf{M}(\mathit{tail}(\sigma))$ otherwise.

Reachable markings: Given a net $N = \langle S, T, F, \mathsf{m}\rangle$, a marking m is *reachable* iff there exists a $\mathsf{fs}\ \sigma \in \mathcal{R}_{\mathsf{m}}^N$ such thats $\mathit{lead}(\sigma)$ is m, and the set of reachable markings of N is $\mathcal{M}_N = \bigcup_{\sigma \in \mathcal{R}_{\mathsf{m}}^N} \mathsf{M}(\sigma)$. Observe that the same marking can be reached with different firing sequences.

States of a net: Given a $\mathsf{fs}\ \sigma = m\,[t_1\rangle\,m_1 \cdots m_{n-1}\,[t_n\rangle\,m'$, with $X_\sigma = \sum_{i=1}^n \{t_i\}$ we denote the multiset of transitions associated to this fs. We call this multiset

a *state* of the net. The set of states of a Petri net is then $\mathsf{St}(N) = \{X_\sigma \in \mu T \mid \sigma \in \mathcal{R}^N_{\mathsf{m}}\}$. Again different firing sequences may give the same state.

Traces: We introduce the notion of trace for a Petri net, which is just the sequence of the labels associated to the transitions in a firing sequence.

Definition 2.3. *Let* $\mathsf{N} = (\langle S, T, F, \mathsf{m}\rangle, l)$ *be a labeled net and let* $\sigma \in \mathcal{R}^N_{\mathsf{m}}$ *be* fs*, with* $\sigma = \mathsf{m}\,[t_1\rangle m_1\,[t_2\rangle m_2 \cdots m_{n-1}\,[t_n\rangle m_n$*, then a* trace *of* N *is the sequence* $l(t_1 t_2 \cdots t_n)$ *and it is denoted with* $run(\sigma)$*.*

Observe that if $\sigma = \mathsf{m}$ then to this fs the empty word is associated, *i.e.* $run(\sigma) = \epsilon$. The set of traces of a net is $\mathsf{Tr}(N) = \{run(\sigma) \mid \sigma \in \mathcal{R}^N_m\}$.

Safe nets: A net is said *safe* whenever the flow relation has value in $\{0, 1\}$ and its places hold at most one token in all possible evolutions.

Definition 2.4. *A Petri net* $N = \langle S, T, F, \mathsf{m}\rangle$ *is said* safe *if* $[\![F]\!] = F$ *and each marking* $m \in \mathcal{M}_N$ *is such that* $m = [\![m]\!]$*.*

When it is not stated differently, we will consider only safe nets $N = \langle S, T, F, \mathsf{m}\rangle$ where each transition can be fired, *i.e.* $\forall t \in T.\ \exists m \in \mathcal{M}_N.\ m\,[t\rangle$.

Subnet: A subnet of a net is a net obtained restricting places and transitions, and correspondingly also the multirelation F and, possibly, the initial marking.

Definition 2.5. *Let* $N = \langle S, T, F, \mathsf{m}\rangle$ *be a Petri net and let* $T' \subseteq T$*. Then the subnet generated by* T' *is the net* $N|_{T'} = \langle S', T', F', \mathsf{m}'\rangle$*, where*

- $S' = \bigcup_{t \in T'} \left([\![{}^\bullet t]\!] \cup [\![N^\bullet t]\!]\right) \ \cup\ \{s \in S \mid \mathsf{m}(s) > 0\}$,
- F' *is restriction of* F *to* S' *and* T'*, and*
- m' *is the multiset on* S' *obtained by* m *restricting to places in* S'*.*

Observe that $N|_{T'}$ may have isolated places and it may be not connected. Analogously we can restrict the net to a subset of places.

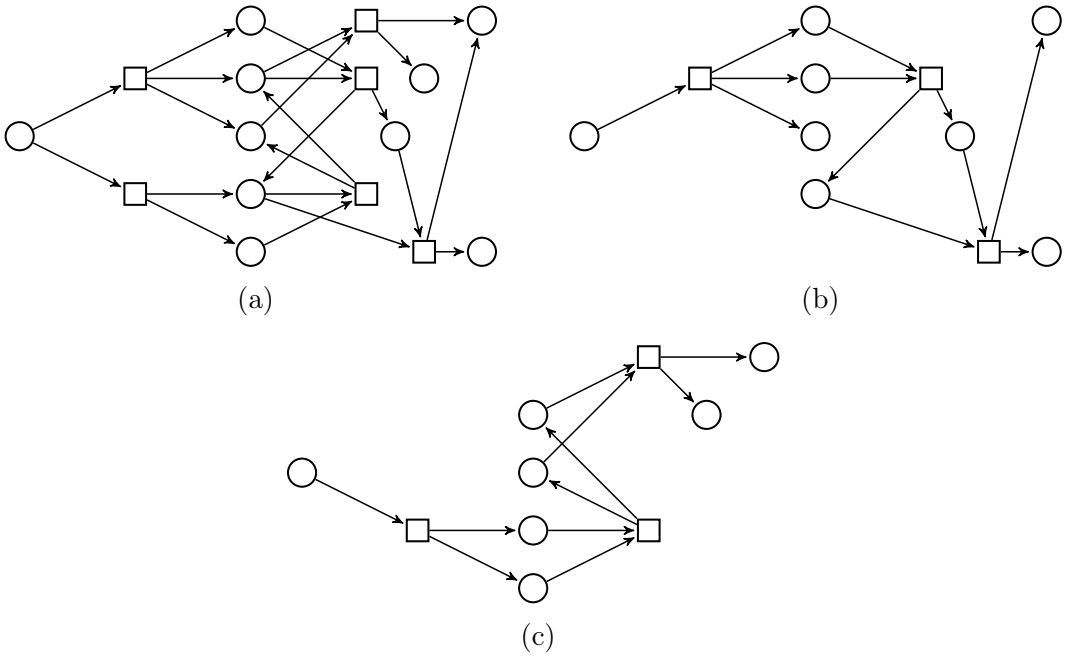


Figure 2.1: A net and two of its subnets.

Definition 2.6. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ be a Petri net and let $S' \subseteq S$. Then the subnet generated by S' is the net $N|_{S'} = \langle S', T', F', \mathsf{m}'\rangle$, where*

- $T' = \{t \in T \mid F(t,s) > 0$ *or* $F(s,t) > 0$ *for* $s \in S'\}$,
- F' *is restriction of* F *to* S' *and* T',
- m' *is the multiset on* S' *obtained by* m *restricting to places in* S'.

Acyclicity: Given a net $N = \langle S, T, F, \mathsf{m}\rangle$, we can associate to it a relation $\leqslant_N$ associated to the flow relation and defined as the reflexive and transitive closure of the relation $x <_N y$ iff $F(x,y) \geqslant 0$.

Definition 2.7. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ be a Petri net and let $S' \subseteq S$ and $T' \subseteq T$. N is said to be* acyclic *with respect to S' and T' whenever $\leqslant_N$ is a partial order, where $\leqslant_N$ is the transitive and reflexive closure of $<_N \cap (S' \cup T') \times (S' \cup T')$.*

We say that $N = \langle S, T, F, \mathsf{m}\rangle$ is acyclic if it is acyclic with respect to S and T.

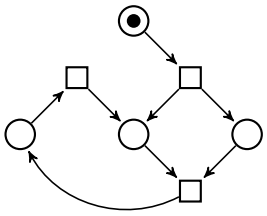


Figure 2.2: An occurrence net.

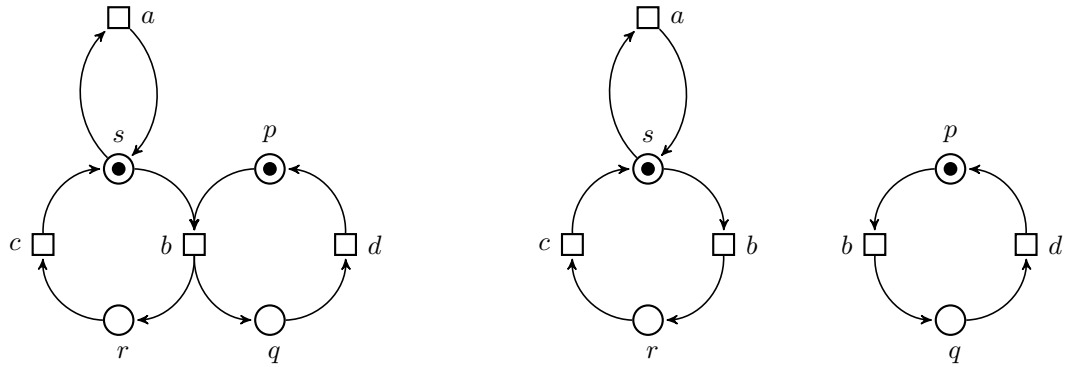


Figure 2.3: A multi-clock net and its two components.

Occurrence Nets: The notion of occurrence net we use here is the one called 1-occurrence net in [vP09] and the intuition behind it is the following: regardless how tokens are produced or consumed, an occurrence net *guarantees* that each transition can *occur* only once.

Definition 2.8. *An* occurrence net $O = \langle S, T, F, \mathsf{m} \rangle$ *is a Petri net where each state is a set, i.e.* $\forall X \in \mathsf{St}(O)$ *it holds that* $X = [\![X]\!]$.

Example 2.1. *An example of occurrence net is shown in Fig 2.2.*

Multi-clock nets: Safe nets can be seen as formed by various *sequential* components (automata) synchronizing on common transitions. This intuition is formalized in the notion of *multi-clock* nets, introduced by E. Fabre in [Fab07].

Definition 2.9. *A* multi-clock *net* N *is a safe net* $\langle S, T, F, \mathsf{m} \rangle$ *such that there exists a mapping* $\nu : S \to \mathsf{m}$ *such that*

- *for all* $s, s' \in [\![m]\!]$, *it holds that* $s \neq s'$ *implies* $\nu^{-1}(s) \cap \nu^{-1}(s') = \varnothing$,

- $\bigcup_{s\in[\![m]\!]} \nu^{-1}(s) = S$,
- $\nu|_{\mathsf{m}}$ *is the identity, and*
- *for all* $t \in T$. ν *is injective on* $[\![{}^\bullet t]\!]$ *and on* $\mathsf{t}^\bullet$, *and* $\nu({}^\bullet\mathsf{t}) = \nu(\mathsf{t}^\bullet)$.

Given a safe net N it can easily turned in a multi-clock one by adding *complementary* places to all the places of the net beside the ones involved in self loop and defining the trivial partition mapping as the one mapping the unmarked complementary place to the marked original one and each unmarked place to the marked complementary one. It is trivial to observe that the partition mapping is not necessarily unique. We will often denote multi-clock nets as a pair explicitly adding the partition mapping: (N, ν), and the partition mapping will be denoted with $\nu(N)$

Given $s \in S$, with $\overline{s}$ we denote the subset of places defined by $\nu^{-1}(\nu(s))$. The consequences of the two requirements, namely (a) $\nu|_{\mathsf{m}}$ is the identity and (b) ν is injective on the preset (postset) of each transition, is that for each $s \in \mathsf{m}$, the net $\langle S, T, F, \mathsf{m}\rangle|_{\overline{s}} = \langle \overline{s}, T_{\overline{s}}, F_{\overline{s}}, \mathsf{m}_{\overline{s}}\rangle$ is a state-machine net, *i.e.* the preset and the postset of each transition has at most one element. State-machine nets can be considered as finite state automata, and the net $\langle S, T, F, \mathsf{m}\rangle$ can be seen as the *union* of the various components.

Example 2.2. *Consider the net in figure 2.3, the two partitions are identified by the following partition mapping* $\nu(s) = s$, $\nu(r) = s$, $\nu(p) = p$ *and* $\nu(q) = p$.

2.3 Causal nets and prime event structure

The notion of occurrence net is a *semantical* one, as it requires that the states of the net enjoy a suitable property, whereas the one of *causal net* is much more syntax oriented. For denoting places and transitions of a causal net we use B and E (see [Win87]) and call them conditions and events respectively. A causal net

is acyclic, when the whole set of conditions is considered, and equipped with a *conflict* relation.

Definition 2.10. *A* causal net $C = \langle B, E, G, \mathsf{c}\rangle$ *is a safe net satisfying the following restrictions:*

1. $\forall b \in [\![\mathsf{c}]\!],\ {}^\bullet b = \varnothing$,
2. $\forall b \in B.\ \exists b' \in [\![\mathsf{c}]\!]$ *such that* $b' \leqslant_C b$,
3. $\forall b \in B.\ {}^\bullet b$ *is either empty or a singleton,*
4. *for all* $e \in E$ *the set* $\{e' \in E \mid e' \leqslant_C e\}$ *is finite, and*
5. $\#$ *is an irreflexive and symmetric relation defined as follows:*
   a) $e \#_r e'$ *iff* $e, e' \in E$, $e \neq e'$ *and* ${}^\bullet e \cap {}^\bullet e' \neq \varnothing$,
   b) $x \# x'$ *iff* $\exists y, y' \in E$ *such that* $y \#_r y'$ *and* $y \leqslant_C x$ *and* $y' \leqslant_C x'$.

The intuition behind this notion is the following: each condition b represents the occurrence of a token, which is produced by the *unique* event in ${}^\bullet b$, unless b belongs to the initial marking, and it is used by only one transition (hence if $e, e' \in b^\bullet$, then $e \# e'$). Furthermore each event has a finite number of predecessors and the immediate conflict relation, stipulating that two events are in conflict if they *compete* on a common resource, is inherited along the flow relation.

On causal net it is natural to define a notion of *causality* among elements of the net: we say that x is *causally dependent* from y iff $y \leqslant_C x$. Given a causal net $C = \langle B, E, G, \mathsf{c}\rangle$, if $\forall b \in B$ it holds that $b^\bullet$ is at most a singleton, we say that it is a *conflict-free* causal net (the relation $\#$ is empty). The following proposition is obvious.

Proposition 2.1. *Let* $C = \langle B, E, G, \mathsf{c}\rangle$ *be a causal net. Then* C *is also an occurrence net.*

Definition 2.11. *Let* $C = \langle B, E, G, \mathsf{c}\rangle$ *be a causal net. We say that* $C' = \langle B', E', G', \mathsf{c}'\rangle$ *if a prefix of* C $(C \leqslant C')$ *whenever*

- $\mathsf{c} = \mathsf{c}'$,
- $E' \subseteq E$,
- $B' = E'^{\bullet} \cup {}^{\bullet}E' \cup \mathsf{c}$,
- $G'(x, y) = 1$ *iff* $G(x, y) = 1$, *with* $x, y \in B' \cup E'$, *and*
- $\forall b \in B', \forall x \leqslant_G b$ *it holds* $x \in B' \cup E'$.

It follows that:

Proposition 2.2. *Let* C *be a causal net and let* C' *be a net such that* $C' \leqslant C$. *Then* C' *is a causal net.*

Prime event structures (PES) [NPW81, Win87] are a simple event-based model of concurrent computations in which events are considered as atomic and instantaneous steps, which can appear only once in a computation. The relationships between events are expressed by two binary relations: *causality* and *conflict*. The relevance of the notion of prime event structure is rooted in the well known relation with another central notion for modeling computations, namely the one of *domain*.

Definition 2.12. *A* prime event structure (PES) *is a tuple* $P = (E, \leqslant, \#)$, *where* E *is a set of* events *and* $\leqslant$, $\#$ *are binary relations on* E *called* causality relation *and* conflict relation *respectively, such that:*

1. *the relation* $\leqslant$ *is a partial order and the set* $\lfloor e \rfloor = \{e' \mid e' \leqslant e\}$ *is finite for all* $e \in E$, *and*
2. *the relation* $\#$ *is irreflexive, symmetric and hereditary with respect to* $\leqslant$, i.e., $e \# e'$ *and* $e' \leqslant e''$ *imply* $e \# e''$ *for all* $e, e', e'' \in E$.

An event can occur only after some other events (its causes) have taken place, and the execution of an event can prevent the execution of other events. This is formalized via the notion of *configuration* of a PES $P = \langle E, \leqslant, \# \rangle$, which is a subset of events $C \subseteq E$ such that for all $e, e' \in C$ $\neg(e \# e')$ (*conflict-freeness*) and $\lfloor e \rfloor \subseteq C$ (*left-closedness*).

Causal nets and PES are closely related: let $C = \langle B, E, G, \mathsf{m} \rangle$ be a causal net. Then $\mathcal{E}_{\text{PES}}(C) = (E, \leqslant, \#)$ is a PES, where $\leqslant = \leqslant_C \cap (E \times E)$ and $\#$ are the causality and conflict relations obtained by the causal net (see [Win87]). To a configuration of the associated PES it is possible to associate a marking in the causal net.

Proposition 2.3. *Let $C = \langle B, E, G, \mathsf{c} \rangle$ be a causal net, and let $X \subseteq E$ be a configuration of $\mathcal{E}_{\text{PES}}(C) = (E, \leqslant, \#)$. Then $X \in \mathsf{St}(C)$ and, given $\sigma \in \mathcal{R}^C_{\mathsf{m}}$ such that $X = X_\sigma$, $\mathbf{mark}(X) = lead(\sigma)$ is the marking reached executing the events in X.*

We observe that, given a configuration X of the PES associated to a causal net C, the subnet $C|_X$ is a causal conflict-free net, *i.e.* each condition b is such that $[\![b^\bullet]\!]$ is a singleton.

2.4 Unfoldings and branching processes

The behavior of a Petri net can be described in many ways, *e.g* using the *marking graph*, or the set of *firing sequences*, or its *unfolding* (see [DR15, Rei13] among many others). The notion of unfolding ([Win87, Eng91]) is particularly relevant as it allows to record conflicts and dependencies among the activities modeled with a Petri net, and the possibility of finding a finite representation of it (the prefix), has given profitability to the notion, otherwise confined to the purely theoretical modeling realm ([McM93, ERV02]).

Given a net $N = \langle S, T, F, \mathsf{m} \rangle$, the unfolding is a labeled causal net where the labeling enjoys some additional requirement.

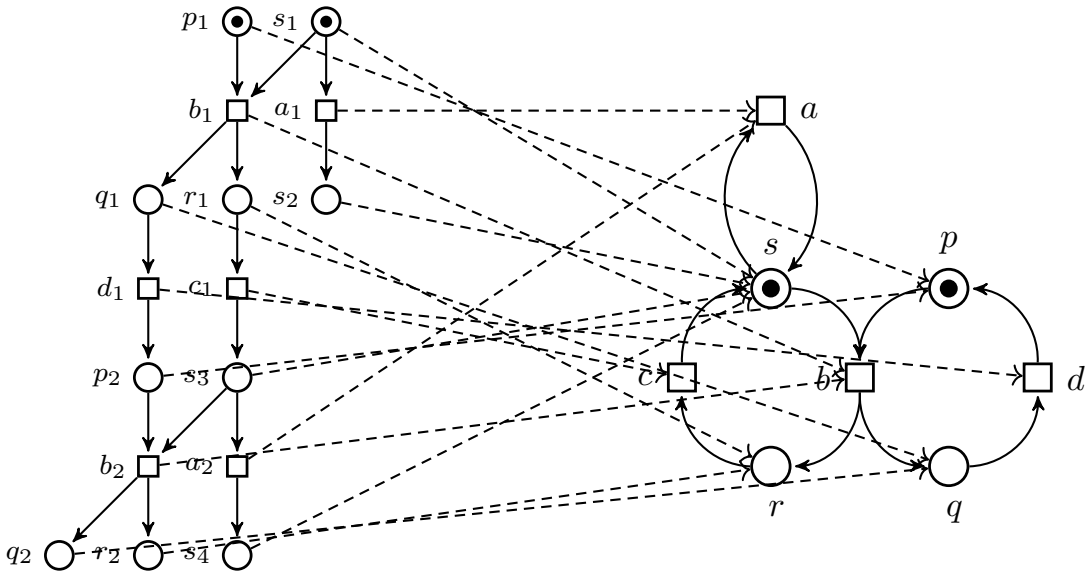


Figure 2.4

Definition 2.13. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ be a safe net and let $\mathsf{C} = (C, p)$ be labeled causal net, with $C = \langle B, E, G, \mathsf{c}\rangle$, where $p : B \cup E \to S \cup T$ is a labeling mapping such that*

- *$p(B) \subseteq S$, $p(E) \subseteq T$,*
- *there are bijections between ${}^\bullet e$ and ${}^\bullet p(e)$, $e^\bullet$ and $p(e)^\bullet$,*
- *there is a bijection between c and m, and*
- *for all $e, e' \in E$, if ${}^\bullet e = {}^\bullet e'$ and $p(e) = p(e')$ then $e = e'$.*

We call C a branching process *of N, and we call p a* folding.

It is shown in [Eng91] that there is a unique maximal branching process, w.r.t. prefix relation. The uniqueness is up to isomorphism. It is called *unfolding* of the net. For a complete overview of branching processes and unfoldings see ([Win87] and [Eng91]).

Example 2.3. *Consider the nets in Fig. 2.4. The one on the left is a branching process of the one on the right. The dashed lines represent the images of the conditions and events of the branching process through the labeling p. Each event*

of the branching process represents a transition of the safe net, and each condition represents a token *in a place of the net.*

The causal net represents the non sequential behaviour of the safe net, and in particular the followings hold (let C be the unfolding of N):

- to each reachable markings $m \in \mathcal{M}_N$ there exists a reachable marking $m' \in \mathcal{M}_C$ such that $p(m') = m$,
- each reachable marking $m \in \mathcal{M}_C$ is mapped to a reachable marking of N, and
- for each reachable markings $m \in \mathcal{M}_C$ and for each $e \in E$ such that $m\,[e\rangle$, there exist a marking $m'' \in \mathcal{M}_N$ and a transition t of N such that $p(m)\,[t\rangle\, m''$, $p(e) = t$ and and $p(m') = m''$, where m' is the marking reached executing e, *i.e.* $m\,[e\rangle\, m'$.

Since a net can have one (or more) infinite firing sequence(s), like the one in Fig. 2.4, the resulting unfolding is, in general, an infinite causal net.

A folding can be seen as a morphism between nets. We recall a definition of morphism that applies in this case.

Definition 2.14. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ and $N' = \langle S', T', F', \mathsf{m}'\rangle$ be two nets. A* morphism *$h : N \to N'$ is a pair $\langle h_T, h_S\rangle$, where $h_T : T \to T'$ is a partial function and $h_S \subseteq S \times S'$ is a relation such that*

- *for each $s' \in \mathsf{m}'$ there exists a unique $s \in \mathsf{m}$ and $s\ h_S\ s'$,*
- *if $s\ h_S\ s'$ then the restriction $h_T\colon {}^\bullet s \to {}^\bullet s'$ and $h_T\colon s^\bullet \to s'^\bullet$ are total functions, and*
- *if $t' = h_T(t)$ then $h_S^{op}\colon {}^\bullet t' \to {}^\bullet t$ and $h_S^{op}\colon t'^\bullet \to t^\bullet$ are total functions, where h_S^{op} is the opposite relation to h_S.*

If $s \in S$ we denote with $h_S(s)$ the set $\{s' \in S' \mid s\ h_S\ s'\}$ and, if $t \in T$, with $h_T(t)$ the set $\{t' \in T' \mid h_T(t) = t'\}$.

Morphisms between nets enjoy the following property:

Proposition 2.4. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ and $N' = \langle S', T', F', \mathsf{m}'\rangle$ be two nets, let $\langle h_T, h_S\rangle \colon N \to N'$ be a morphism, and let $t \in T$ be a transition. Then*

a) ${}^\bullet h_T(t) = h_S({}^\bullet t)$

b) $h_T(t)^\bullet = h_S(t^\bullet)$

Proof. *a)* Let $s \in {}^\bullet h_T(t)$. Consider the set $h_S({}^\bullet t) = \{s' \in S' \mid \exists \bar{s} \in {}^\bullet t.\ \bar{s}\ h_S\ s'\}$. By definition of morphism $h_S^{op} \colon {}^\bullet h_T(t) \to {}^\bullet t$ is a total function, hence there exists $s' \in {}^\bullet t$ such that $s\ h_S\ s'$. Thus ${}^\bullet h_T(t) \subseteq h_S({}^\bullet t)$. Take now $s \in {}^\bullet t$ and consider s' such that $s\ h_s\ s'$, *i.e.* $s' \in h_S({}^\bullet t)$. Now, $h_T \colon s^\bullet \to s'^\bullet$ is a total function, and $h_T(t) \in s'^\bullet$. Hence $s' \in {}^\bullet h_T(t)$. But this means that $h_S({}^\bullet t) \subseteq {}^\bullet h_T(t)$.

b) Let $s \in h_T(t)^\bullet$. Consider again the set $h_S(t^\bullet) = \{s' \in S' \mid \exists \bar{s} \in t^\bullet.\ \bar{s}\ h_S\ s'\}$. By definition of morphism $h_S^{op} \colon h_T(t)^\bullet \to t^\bullet$ is a total function, hence there exists $s' \in t^\bullet$ such that $s\ h_S\ s'$. Thus $h_T(t)^\bullet \subseteq h_S(t^\bullet)$. Take now $s \in t^\bullet$ and consider s' such that $s\ h_S\ s'$, *i.e.* $s' \in h_S(t^\bullet)$. Now, $h_T \colon {}^\bullet s \to {}^\bullet s'$ is a total function, and $h_T(t) \in {}^\bullet s'$. Hence $s' \in h_T(t)^\bullet$. But this means that $h_S(t^\bullet) \subseteq h_T(t)^\bullet$. ■

Net morphisms preserve the firing of transitions.

Theorem 2.1. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ and $N' = \langle S', T', F', \mathsf{m}'\rangle$ be two nets, let $\langle h_T, h_S\rangle \colon N \to N'$ be a morphism, let $m, m' \in \mathcal{M}_N$ be two markings of N, and $t \in T$. If $m\,[t\rangle\, m'$, then $h_S(m)\,[h_T(t)\rangle\, h_S(m')$.*

Proof. As $m\,[t\rangle\, m'$ we know that ${}^\bullet t \subseteq m$. Using Prop. 2.4 we have that $h_S({}^\bullet t) = {}^\bullet h_T(t)$, hence ${}^\bullet h_T(t) \subseteq h_S(m)$. Using the same proposition we have also that $h_S(t^\bullet) = h_T(t)^\bullet$, thus $h_S(m') = h_S(m) - {}^\bullet h_T(t) + h_T(t)^\bullet$. ■

Theorem 2.1 can be trivially lifted to firing sequences, as the following corollary shows.

Corollary 2.1. *Let N and N' be two nets, let $h = \langle h_T, h_S \rangle \colon N \to N'$ be a net morphism, let $m, m_1, \dots m_n \in \mathcal{M}_N$, let $t_1, \dots, t_n$ a set of transitions of N, and let $\sigma = m\,[t_1\rangle\, m_1 \dots m_{n-1}\,[t_n\rangle\, m_n$ be a firing sequence,* i.e. *$\sigma \in \mathcal{R}^N$. Then also $h(\sigma) = h_S(m)\,[h_T(t_1)\rangle\, h_S(m_1) \dots h_S(m_{n-1})\,[h_T(t_n)\rangle\, h_S(m_n)$ is a firing sequence and $h(\sigma) \in \mathcal{R}^{N'}$.*

As a consequence of Th. 2.1 and Cor. 2.1 we have that net morphisms preserve firing sequences and thus they preserve reachable markings.

The composition of two net morphisms is again a net morphism.

Let h_S and h'_S two relations, $h_S \circ h'_S \subseteq S \times S''$ is the relation defined as $s\ h_S \circ h'_S\ s''$ iff $\exists s'$ such that $s\ h_S\ s'$ and $s'\ h'_S\ s''$.

Definition 2.15. *Let $\langle h_T, h_S \rangle \colon N \to N'$,$\langle h'_T, h'_S \rangle \colon N' \to N''$, $h = \langle h_T, h_S \rangle$ be two nets morphisms. We define their* composition *as $\langle h'_S \circ h_S, h'_T \circ h_T \rangle$.*

Proposition 2.5. *Let $N = \langle S, T, F, \mathsf{m} \rangle$, $N' = \langle S', T', F', \mathsf{m}' \rangle$, and $N'' = \langle S'', T'', F'', \mathsf{m}'' \rangle$ be three nets. Let $\langle h_T, h_S \rangle \colon N \to N'$, $\langle h'_T, h'_S \rangle \colon N' \to N''$, let $h = \langle h_T, h_S \rangle$ be two net morphisms. Then $\langle h'_S \circ h_S, h'_T \circ h_T \rangle$ is a morphism from N to N''.*

Proof. Take $s'' \in \mathsf{m}''$. As $h' = \langle h'_T, h'_S \rangle$ is a morphism, there exist a unique $s' \in \mathsf{m}'$ such that $s'\ h'_S\ s''$. Also $h = \langle h_T, h_S \rangle$ is a morphism, hence there exists a unique $s \in m$ such that $s\ h_S\ s'$. Hence there exists a unique $s\ h_S \circ h'_S\ s''$.

Take now $s\ h_S \circ h'_S\ s''$. There exists $s' \in S'$ such that $s\ h_S\ s'$ and $s'\ h'_S\ s''$. We have that both $h_T \colon {}^\bullet s \to {}^\bullet s'$ and $h'_T \colon {}^\bullet s' \to {}^\bullet s''$ are total function, ad h and h' are morphisms, hence also $h'_T \circ h_T \colon {}^\bullet s \to {}^\bullet s''$ is total. Similarly for $h'_T \circ h_T \colon s^\bullet \to s''^\bullet$.

Finally consider $t'' = h'_T(h_T(t))$. As h' is a morphism, $h'^{op}_S \colon {}^\bullet t'' \to {}^\bullet h_T(t)$ is a total function, and, as h is a morphism, also $h^{op}_S \colon {}^\bullet h_T(t) \to {}^\bullet t$ is a total function. Their composition $(h'_S \circ h_S)^{op} = h^{op}_S \circ h'^{op}_S \colon {}^\bullet t'' \to {}^\bullet t$ is a total function as well. By reasoning in the same way we have that also $(h'_S \circ h_S)^{op} \colon t''^\bullet \to t^\bullet$ is total. This concludes the proof. ■

Given a branching process $\mathsf{C} = (C, p)$ of a safe net $N = \langle S, T, F, \mathsf{m}\rangle$, where $C = \langle B, E, G, \mathsf{c}\rangle$, with define the morphism $h^p \colon C \to N$ as follows:

- $h^p_T \colon E \to T$ is $p|_E$, and
- $(b, s) \in h^p_S$ iff $p(b) = s$

Proposition 2.6. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ and let $\mathsf{C} = (C, p)$ be a branching process of N. Then $h^p \colon C \to N$ is a net morphism.*

Proof. Clearly if $b \in C$ there is just one $s \in m$ such that $p(b) = s$, as there is a bijection between c and m.

As there are bijection between ${}^\bullet\mathsf{e}$ and ${}^\bullet p(\mathsf{e})$, $\mathsf{e}^\bullet$ and $p(\mathsf{e})^\bullet$, we have that $(h^p_S)^{op} \colon {}^\bullet p(\mathsf{e}) \to {}^\bullet\mathsf{e}$ and $(h^p_S)^{op} \colon p(\mathsf{e})^\bullet \to \mathsf{e}^\bullet$ are total.

Finally consider b, $p(b)$, and $h^p_T \colon {}^\bullet b \to {}^\bullet p(b)$. As p is a labeling, it is total, hence the thesis. ■

2.5 Finite Prefix

In this section we briefly recall one the main techniques that permits to cope with infinite structures like unfoldings.

Definition 2.16. *Let $\mathsf{C} = (C, p)$ and $\mathsf{C}' = (C', p')$ two branching processes of a net N. Then C' is a* prefix *of C if $C_{|C'}$ satisfies:*

1) if $b' \in B'$, then $\forall e \in {}^\bullet b'$ such that $e \in E$ it holds that $e \in E'$,

2) if $e' \in E'$, then $\forall b \in {}^\bullet e'$ such that $b \in B$ and $\forall b \in e'^\bullet$ such that $b \in B$, it holds that $b \in B$, and

3) p' is the restriction of p to conditions and events in C'.

Although unfoldings, in general, cannot be easily treated (due to the existence of infinite runs in the original net), it is possible to identify a suitable finite subnet

of an unfolding where all the relevant informations about sependencies and conflicts are preserved: the *complete prefix*.

Complete prefixes have been introduced in [McM93] where a general algorithm for truncating nets is presented along the applications of prefixes to model checking. Subsequently, in [ERV02], the authors refines the McMillan's algorithm.

We now recall some basic definitions.

Definition 2.17. *Let* $C = \langle B, E, G, \mathsf{c}\rangle$ *be a causal net, two conditions* $b, b' \in B$ *are in* co relation*, denoted with* $b \mathsf{co} b'$*, iff* $\neg(b \leqslant b') \wedge \neg(b' \leqslant b) \wedge \neg(b \# b')$.

In this definition the conflict relation and the partial order are those of the causal net C.

A set of conditions of a branching process is a *co-set* if its elements are pairwise in co relation. A maximal co set with respect to set inclusion is called a *cut*.

Definition 2.18. *Let* C *be a branching process. A finite configuration* X *of* C *is a subset of events which is* conflict-free *and* left closed, i.e. *if* $e \in C$ *and* $e' \leqslant e$ *then* $e' \in C$.

From now on we will consider finite configurations only (thus the adjective *finite* will be omitted). Let X be a configuration of $C = \langle B, E, G, \mathsf{c}\rangle$, we denote with $\mathbf{Cut}(X)$ the set

$$\mathbf{Cut}(X) = (\mathsf{c} \cup X^\bullet) \setminus {}^\bullet X$$

$\mathbf{Cut}(X)$ is a cut that includes all conditions marked by executing the events in X and the remaining conditions of the initial marking c not consumed by X. $\mathbf{Cut}(X)$ represents a reachable marking of the original net ($p(\mathbf{Cut}\, C)$), and we denoted the latter with $\mathbf{M}(X)$.

Definition 2.19. *A branching process* C *of a net* N *is said* complete *if for every reachable marking* $m \in \mathcal{M}_N$ *there exists a configuration* X *in* C *such that:*

1) $\mathbf{M}(X) = m$*, and*

2) for every transition t enabled at m ($m\,[t\rangle$) there exists a configuration $X \cup \{\mathsf{e}\}$ such that $\mathsf{e} \notin X$ and e is labeled by t.

Condition 1) says that every marking of N must be represented in C and condition 2) is the so called *preservation of firings*, it means that a branching process must contain at least one instance of all events that can be fired from a marking M.

The number of reachable markings of a safe Petri net is finite so there always exists a finite complete branching process.

Chapter 3

Unravel nets

In this chapter we introduce a new class of nets, *unravel nets*, which can be considered as a generalization of causal nets. The idea is rather simple: in each executions it must be possible to extract dependencies among transitions syntactically, whereas the conflict is confined to the semantic level, as the syntactically definable conflicts are just a part of the whole set of conflicts.

We do two sanity checks on this notion. First we show that a causal net is indeed an unravel net (which is rather obvious) and then we will show that unravel nets are tightly related to *bundle event structure* (BES) [Lan92, Lan93]. We will show how, given an unravel net, it is possible to associate a bundle event structure, and how to construct an unravel net from a bundle event structure, in such a way that the states of an unravel net and the configurations of the related BES coincide.

Causal nets (Def. 2.10 of subsection 2.3) capture dependencies and conflicts among transitions (events) whereas occurrence nets capture the unique occurrence property of each transition. Dependencies are inferred using the partial order obtained from the flow relation whereas conflicts are obtained using the notion of immediate conflicts (two events share a condition in their preset) and adding conflicts involving all the events that depends on the initial ones.

We define a net which will turn to be, so to say, in between occurrence and causal nets. Like in occurrence nets we require that each transition happens just

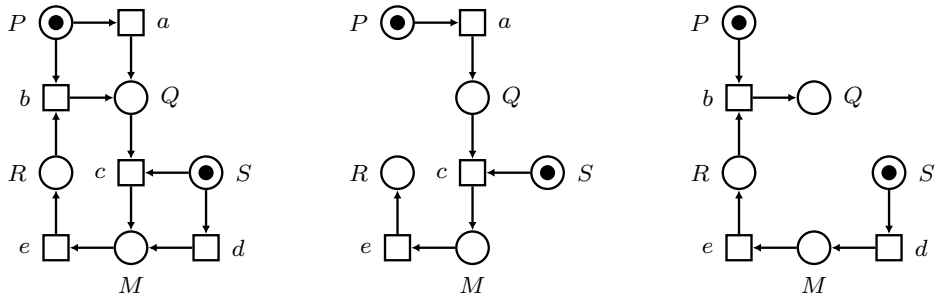


Figure 3.1: An unravel but not causal net, and two of its subnets (causal nets).

once in a configuration, and similarly to causal nets we still want to be able to retrieve dependencies among the firings of transitions, though in a more *semantical way.*

Definition 3.1. *An* unravel net $R = \langle S, T, F, \mathsf{m}, l \rangle$ *is an occurrence net such that*

a) R is safe,

b) for each state $X \in \mathsf{St}(R)$ the net $R|_{[\![X]\!]}$ is a conflict-free causal net, and

c) for each transition $t \in T$ there exists a state $X \in \mathsf{St}(R)$ such that $t \in X$.

The idea here is that if we focus on each single execution, the restriction of the net to the events in this execution is an acyclic net where each condition has at most one incoming arc and one outgoing arc (thus is a conflict free causal net). The fact that a safe net is an occurrence net can be easily enforced by adding, for each transition t a place s_t in the preset of t, marking it initially and such that ${}^\bullet s_t = \varnothing$. The last condition has two consequences. One is that each transition of the net may really represents the firing of a transition in the net of which the unravel net may represent the behavior, and the second one is that initial conditions may be easily identified as they have an empty preset.

We first show that unravel nets are a conservative extension of the notion of causal net. Indeed, it is straightforward to observe that if $C = \langle B, E, G, \mathsf{c} \rangle$ is a causal net then it is an unravel net as well.

Proposition 3.1. *Let $C = \langle B, E, G, \mathsf{c} \rangle$ be a causal net. Then C is an unravel net as well.*

Proof. The first two conditions are trivial. It remains to prove that for each transition $e \in E$ there exists a state $X \in \mathsf{St}(C)$ such that $e \in X$. Consider $\lfloor e \rfloor = \{e' \in E \mid e' \leqslant_C e\}$, it is trivial to observe that $\lfloor e \rfloor \in \mathsf{St}(C)$, hence the thesis. ■

The contrary does not hold as the following example shows:

Example 3.1. *Consider the net in figure 3.1a. Clearly it is not a causal net, as the place Q has two incoming arcs. The states of the net are $\{a\}, \{d\}, \{a, c\}, \{a, d\}$, $\{d, e\}, \{a, d, e\}$, $\{a, c, e\}, \{d, e, b\}$. The net is safe, and each state induces a conflict-free causal net so, by definition, it is an unravel net. Figures 3.1b and 3.1c depict the nets that correspond to states $\{a, c, e\}$ and $\{d, e, b\}$.*

We list some properties of unravel nets. First of all, like in causal nets, places in the initial marking have no incoming arc.

Proposition 3.2. *Let $R = \langle S, T, F, \mathsf{m} \rangle$ be an unravel net and let $s \in S$ such that $s \in [\![\mathsf{m}]\!]$ then ${}^\bullet s = \varnothing$.*

Proof. Assume ${}^\bullet s \neq \varnothing$, then there exists a transition $t \in T$ such that $t \in {}^\bullet s$. Now, as R is an unravel net, t can be executed, hence there exists a fs σ such that $\sigma\,[t\rangle\, m'$. we have two cases:

a) each transition t' in σ is such that $s \notin {}^\bullet t'$, but then $m'(s) = 2$, contradicting the safeness of R,

b) there is a transition t' in σ such that $s \in {}^\bullet t'$, but then $R|_{[\![\mathcal{X}_\sigma]\!]}$ is cyclic, contradicting the hypothesis that R is an unravel net. ■

The main difference between unravel nets and causal nets is that conflicts among transitions in unravel nets have to be defined *semantically* whereas in causal nets they are syntax driven.

Definition 3.2. *Let $R = \langle S, T, F, \mathsf{m}\rangle$ be an unravel net. We define the* conflict *relation $\# \subseteq T \times T$ as follows: $t \# t'$ whenever $t \neq t'$ and $\forall X \in \mathsf{St}(R)$ it holds that $\{t, t'\} \not\subseteq X$.*

Though the definition of conflict is a semantic one, some conflicting transitions can be characterized syntactically.

Proposition 3.3. *Let $R = \langle S, T, F, \mathsf{m}\rangle$ be an unravel net, let $s \in S$ and $t, t' \in T$. If $t^\bullet \cap t'^\bullet \neq \varnothing$ then $t \# t'$.*

Proof. Let $s \in t^\bullet \cap t'^\bullet$. Assume that $\neg(t \# t')$, then exists a fs σ such that $\sigma = \sigma' \left[t\right\rangle \sigma'' \left[t'\right\rangle \sigma'''$ with

$$\sigma'' = \begin{cases} \bar{m} \text{ or} \\ \bar{m} \left[\bar{t}\right\rangle \bar{\sigma} \end{cases}$$

or $\sigma = \sigma' \left[t'\right\rangle \sigma'' \left[t\right\rangle \sigma'''$ with

$$\sigma''' = \begin{cases} \hat{m} \text{ or} \\ \hat{m} \left[\hat{t}\right\rangle \hat{\sigma} \end{cases}$$

Then $lead(\sigma' \left[t\right\rangle \bar{m})(s) = 1$ and $lead(\sigma' \left[t\right\rangle \sigma'' \left[t'\right\rangle \hat{m}) \geqslant 1$. If $lead(\sigma' \left[t\right\rangle \sigma'' \left[t'\right\rangle \hat{m}) \geqslant 2$ we violate the safeness, if $lead(\sigma' \left[t\right\rangle \sigma'' \left[t'\right\rangle \hat{m}) = 1$ the net has a cycle. ■

Proposition 3.4. *Let $R = \langle S, T, F, \mathsf{m}\rangle$ be an unravel net, and let $t, t' \in T$. If ${}^\bullet t \cap {}^\bullet t' \neq \varnothing$ then $t \# t'$.*

Proof. Assume that $\neg(t \# t')$ and let $s \in {}^\bullet t \cap {}^\bullet t'$. Then exists a fs σ such that $\sigma' \left[t\right\rangle \sigma'' \left[t'\right\rangle \sigma'''$ or $\sigma' \left[t'\right\rangle \sigma'' \left[t\right\rangle \sigma'''$. With the same argument of Prop. 3.3 we have that s would get marked twice violating the acyclicity of $N|_{X_\sigma}$. ■

3.1 Bundle event structures

Prime event structure are tightly connected with causal nets. We show that a similar relationship exists between unravel nets and *bundle event structures*.

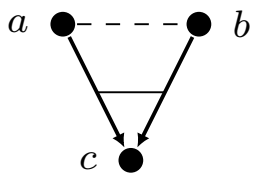


Figure 3.2: A graphical representation of a BES.

In bundle event structures [Lan93] causality is represented by pairs (X, e), the *bundles*, where X is a non empty set of events and e an event. The meaning of a bundle (X, e) is that if e happens then one (and only one) event of X has to have happened before (events in X are pairwise conflicting). An event e can be caused by several bundles, in that case, for each bundle an event in it should have happened.

Definition 3.3. *A* bundle event structure *is a triple* $\beta = (\mathsf{E}, \mapsto, \#)$*, where*

- E *is a set of events,*
- $\#$ *is an irreflexive and symmetric binary relation on* E *(the* conflict *relation),*
- $\mapsto \subseteq \mathbf{2}^{\mathsf{E}}_{fin} \times \mathsf{E}$ *is the* enabling *relation such that if* $X \mapsto \mathsf{e}$ *then for all* $\mathsf{e}_1, \mathsf{e}_2 \in X$*.* $\mathsf{e}_1 \neq \mathsf{e}_2$ *implies* $\mathsf{e}_1 \# \mathsf{e}_2$*, and*
- *for each* $\mathsf{e} \in E$ *it holds that the set* $\bigcup\{X \mid X \mapsto \mathsf{e}\}$ *is finite.*

The final condition is an analogous of the finite cause requirement for prime event structures. Indeed this requirement rules out situations like the following one. Consider an event s such that $\forall i \in \mathbb{N}$ there is a bundle $\{e_i\} \mapsto e$. Then the event e has infinite causes which we want to rule out.

To draw a BES we adopt the usual convention: the conflict relation is depicted as a dotted line, and bundles are depicted as arrows connected together by a straight line.

The configurations of a BES are defined as follows.

Definition 3.4. *Let* $\beta = (\mathsf{E}, \mapsto, \#)$ *be a* BES *and* $X \subseteq \mathsf{E}$ *be a set of events. Then* X *is a* configuration *of* β *iff*

1. *it is* conflict free, i.e. $\forall \mathsf{e}, \mathsf{e}' \in X.\ \mathsf{e} \neq \mathsf{e}' \ \Rightarrow\ \neg(\mathsf{e} \mathrel{\#} \mathsf{e}')$, *and*

2. *there exists a linearization* $\{\mathsf{e}_1, \ldots, \mathsf{e}_n, \ldots\}$ *of the events in* X *such that* $\forall i \in \mathbb{N}$ *and for all bundles* $X_{j_i} \mapsto \mathsf{e}_i$ *it holds that* $X_{j_i} \cap \{\mathsf{e}_1, \ldots, \mathsf{e}_{i-1}\} \neq \varnothing$.

The requirements are the usual ones: it must be conflict free and each event must have all of its causes. The causes of an event in a BES, as said before, have to be chosen using all the bundles involving the event.

We first show that BES are a conservative extension of PES. Indeed each PES $P = (\mathsf{E}, \leqslant, \#)$ can be seen as the BES, as the following proposition shows.

Proposition 3.5. *Let* $P = (\mathsf{E}, \leqslant, \#)$ *be a* PES, *then* $\mathcal{B}(P) = (\mathsf{E}, \mapsto, \#)$ *where* $\mapsto = \{(\{\mathsf{e}'\}, \mathsf{e}) \mid \mathsf{e}' < \mathsf{e}\}$ *is a* BES, *and* $\mathsf{Conf}(P) = \mathsf{Conf}(\mathcal{B}(P))$.

Proof. Clearly $\bigcup\{X \mid X \mapsto \mathsf{e}\}$ is finite as $\lfloor \mathsf{e} \rfloor$ is finite, then the bundles of $\mathcal{B}(P)$ are well defined, hence it remains to show that the set of configurations are indeed the same.

Take $X \in \mathsf{Conf}(P)$, then it is clearly conflict free. It remain to show that there exists a linearization $\{\mathsf{e}_1, \ldots, \mathsf{e}_n, \ldots\}$ of the events in X such that $\forall i \in \mathbb{N}$ and for all bundles $X_i \mapsto \mathsf{e}_i$ it holds that $X_i \cap \{\mathsf{e}_1, \ldots, \mathsf{e}_{i-1}\} \neq \varnothing$. For each event e of X, consider a linearization of X compatible with the ordering $\lfloor \mathsf{e} \rfloor$. This linearization clearly satisfies the second condition of Def. 3.4, and then $X \in \mathsf{Conf}(\mathcal{B}(P))$.

For the converse, consider $X \in \mathsf{Conf}(\mathcal{B}(P))$. Again X is conflict free, hence it remains to show that for each $\mathsf{e} \in X$ it holds that $\lfloor \mathsf{e} \rfloor \subseteq X$. As $X \in \mathsf{Conf}(\mathcal{B}(P))$, there exists a linearization $\{\mathsf{e}_1, \ldots, \mathsf{e}_n, \ldots\}$ of elements of X and, given e_i, it holds that each bundle $X_i \mapsto \mathsf{e}_i$ is such that $X_i \cap \{\mathsf{e}_1, \ldots, \mathsf{e}_{i-1}\} \neq \varnothing$. But as the X_i are singletons we have that $X_i \subseteq \{\mathsf{e}_1, \ldots, \mathsf{e}_{i-1}\}$ and as we have a bundle for each event strictly smaller than e_i, it holds that $\lfloor \mathsf{e} \rfloor \subseteq X$, hence $X \in \mathsf{Conf}(P)$. ■

BES are more expressive than PES, as they are able to model *or*-causality. In fact we may have the following BES $\mathsf{a} \mathrel{\#} \mathsf{b}$ (symmetric pair omitted) and $\{\mathsf{a}, \mathsf{b}\} \mapsto \mathsf{c}$ stipulating that the same event may have two different and alternative pasts,

namely one containing a and the other b. The two maximal traces of this BES are ac and bc (see Figure 3.2).

For our purpose finite configurations will suffice, as we will consider mostly finite computations, which are precisely represented by finite configurations. To characterize finite configurations we resort to the notion of *events trace*. Consider a set of events E, $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ is a *sequence* of events in E. With overloading of notation we denote ϵ as the *empty* sequence. Given a sequence $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ of events, with $\overline{\rho}$ we denote the set $\{\mathsf{e}_1, \ldots, \mathsf{e}_n\}$ and clearly the set associated to the empty sequence is the empty set. Given a sequence ρ of events, its length is $|\overline{\rho}|$ and it is denoted with $len(\rho)$ and for each $1 \leqslant i \leqslant len(\rho)$ with ρ_i we denote the sequence $\mathsf{e}_1 \cdots \mathsf{e}_i$, and with ρ_0 we denote the empty sequence.

Definition 3.5. *Let $\beta = (\mathsf{E}, \mapsto, \#)$ be a* BES*. A* trace *is a sequence of distinct events $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ such that*

- $\overline{\rho} \subseteq \mathsf{E}$,
- $\forall 1 \leqslant i, j \leqslant len(\rho).\ \neg(\mathsf{e}_i \ \# \ \mathsf{e}_j)$, *and*
- $\forall 1 \leqslant i \leqslant len(\rho). \forall X \subseteq \mathsf{E}.\ X \mapsto \mathsf{e}_i \Rightarrow \overline{\rho_{i-1}} \cap X \neq \varnothing$.

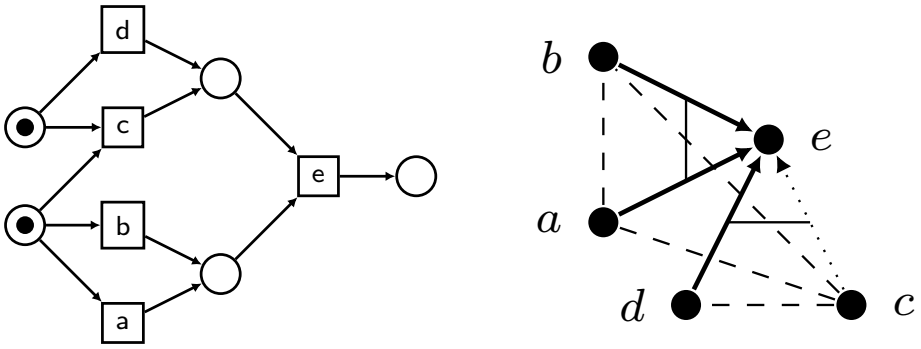
The set of traces of a BES *β is denoted $\mathsf{Tr}(\beta)$.*

Traces, as they are defined, are finite, and they characterize *linearized finite* configurations.

Definition 3.6. *Let $\beta = (\mathsf{E}, \mapsto, \#)$ be a* BES*. Then $X \subseteq_{fin} \mathsf{E}$ is a* finite *configuration iff there exists a trace $\rho \in \mathsf{Tr}(\beta)$ such that $\overline{\rho} = X$. The set of finite configurations of β is denoted with $\mathsf{Conf}_{fin}(\beta)$.*

3.2 Unravel net and BES

In this section we relate unravel nets and BES. We define how to construct a BES from a given unravel net and viceversa. We will relate the states of an unravel net



(a) (b)

Figure 3.3: An unravel net and its associated BES.

to the traces of a BES, showing that there is an 1-1 correspondence.

We first show how to associate a BES to an unravel net. The intuition is rather simple: to each place in the preset of a transition t we associate a bundle X for the correspondent event t in the event structure, and the bundle is formed by the transitions putting a token in that place.

Proposition 3.6. *Let $N = \langle S, T, F, \mathsf{m}\rangle$ be an unravel net, then $\mathcal{E}(N) = (T, \mapsto, \#)$ is a* BES*, where*

- *$t \# t'$ in $\mathcal{E}(N)$ iff $t \# t$ in N, and*
- *for each $t \in T$, for each $s \in {}^\bullet t$, we have ${}^\bullet s \mapsto \mathsf{e}$.*

Proof. We show that $\mathcal{E}(N)$ is indeed an BES. The conflict relation in $\mathcal{E}(N)$ is antisymmetric and irreflexive because it is so in N. Since we are are dealing with unravel nets, all the transitions putting tokens in the same place are in conflict, so the bundles respect Def. 3.3. ■

Example 3.2. *Consider the net in Fig. 3.3a and its associated event structure 3.3b. The bundles are $\{\mathsf{c}, \mathsf{d}\} \mapsto \mathsf{e}$ and $\{\mathsf{a}, \mathsf{b}\} \mapsto \mathsf{e}$, and are obtained syntactically, whereas the conflicts $\mathsf{c} \#\mathsf{d}$, $\mathsf{a} \#\mathsf{b}$, $\mathsf{a} \#\mathsf{c}$ and $\mathsf{b} \#\mathsf{c}$ are obtained using all the possible executions, though in this example they are syntactically deducible as they share a condition in their preset.*

We show that the sets of traces of the unravel net N and of the associated BES $\mathcal{E}(N)$ coincide.

Theorem 3.1. *Let $N = \langle S, T, F, m\rangle$ be an unravel net and $\mathcal{E}(N) = (T, \#, \mapsto)$ be the associated* BES. *Then*

$$\tau \in \mathsf{Tr}(N) \iff \tau \in \mathsf{Tr}(\mathcal{E}(N)).$$

Proof. ($\Leftarrow$) By induction on the length of the trace. For the empty trace it is trivial to observe that the $\mathsf{fs}\ \sigma = m$ is precisely the one we are looking for as $run(\sigma)$ is indeed the empty trace.

Assume it holds for traces of length n. Consider $\tau = t_1 \cdots t_n t_{n+1}$, we know, by induction hypothesis, that $\tau' = t_1 \cdots t_n$ is a trace of N, hence there exists a fs σ such that $run(\sigma) = \tau'$ and $lead(\sigma)$ is the marking reached executing this firing sequence. We show that ${}^\bullet t_{n+1} \subseteq lead(\sigma)$ (hence $lead(\sigma)\,[t_{n+1}\rangle$). Take $s \in {}^\bullet t_{n+1}$, we must show that $lead(\sigma)(s) = 1$. We distinguish two cases:

a) ${}^\bullet s = \varnothing$, then s must belong to the initial marking, otherwise no transition with s as precondition could be executed, contradicting the fact that N is an unravel net. Assume $lead(\sigma)(s) = 0$, there exists $t_j \in \overline{\tau'}$ with $j \leqslant n$ such that $s \in {}^\bullet t_j$, but then, for Prop. 3.4, $t_{n+1} \# t_j$, contradicting the conflict freeness of τ,

b) ${}^\bullet s \neq \varnothing$, then it has been marked by a $t_j \in \overline{\tau'}$ and ${}^\bullet s \mapsto t_{n+1}$. Again assume $lead(\sigma)(s) = 0$, then there exists $t_k \in \overline{\tau'}$ with $j < k \leqslant n$ such that $s \in {}^\bullet t_k$, but then, for Prop. 3.4, $t_k \# t_{n+1}$, contradicting the conflict freeness of τ.

Then $lead(\sigma)\,[t_{n+1}\rangle$ and we have the thesis.

($\Rightarrow$) Assume that τ is a trace of N, then there exists a $\mathsf{fs}\ \sigma$ such that $run(\sigma) = \tau$. To show that it is also a trace of $\mathcal{E}(N)$ we have to prove that

1) it is conflict free and

2) $\forall 1 \leqslant i, j \leqslant len(\tau).\ \neg(\mathsf{e}_i \ \# \ \mathsf{e}_j)$ and $\forall 1 \leqslant i \leqslant len(\tau). \forall Y \subseteq \mathsf{T}.\ Y \mapsto \mathsf{e}_i \Rightarrow \overline{\tau_{i-1}} \cap Y \neq \varnothing$.

Conflict freeness is trivial, by definition of # for unravel nets (Def. 3.1). For 2), we must prove, by induction on the length of τ, that each transition t of τ either has a bundle in $\mathcal{E}(N)$ enabling it or it is such that ${}^\bullet t \subseteq m$. If $len(\tau) = 1$ then $\overline{\tau} = \{t\}$ and the transition t is enabled at initial marking. By Prop. 3.2, each $s \in [\![\mathsf{m}]\!]$ has an empty preset, hence there is no bundle associate to t, thus τ is a trace of $\mathcal{E}(N)$.

Assume that for $n = len(\tau)$ 2) is satisfied.

Consider $\tau t_{n+1} \in \mathsf{Tr}(N)$. We have $lead(\sigma)\,[t_{n+1}\rangle$ where $lead(\sigma)$ is the marking reached executing the firing sequence σ such that $run(\sigma) = \tau$. Thus $\forall s \in {}^\bullet t_{n+1}$, $lead(\sigma)(s) = 1$. We have to prove that each bundle ${}^\bullet s \mapsto t_{n+1}$ is satisfied, for each $s \in {}^\bullet t_{n+1}$. We again distinguish two cases:

1) if ${}^\bullet s = \varnothing$, as there is no bundle associated to it the thesis follows, and

2) if ${}^\bullet s \neq \varnothing$, then s is marked in $lead(\sigma)$ by a transition t_j with $j \leqslant n$ and the bundle ${}^\bullet s \mapsto t_{n+1}$ is satisfied.

It follows that t_{n+1} can be added to τ and $\tau t_{n+1} \in \mathsf{Tr}(\mathcal{E}(N))$. ■

We show now how to associate an unravel net to a BES. We consider only BES where all the events may be executed, namely such that $\forall \mathsf{e} \in \mathsf{E}$ there is a finite configuration X such that $\mathsf{e} \in X$. This assumption is required for generating nets without impossible transitions (see Def. 3.1). The unravel net associated to the event structure $\beta = (\mathsf{E}, \mapsto, \#)$ has as transitions the events of β, and places are defined by bundles, by the conflict relations and, in order to guarantee that the preset of each transition is non empty, also by a place guaranteeing this.

Proposition 3.7. *Let $\beta = (\mathsf{E}, \mapsto, \#)$ be a* BES *such that $\forall \mathsf{e} \in \mathsf{E}\ \exists X \in \mathsf{Conf}_{fin}(\beta).\ \mathsf{e} \in$*

X. Then $\mathcal{N}(\beta) = \langle S, E, \mathrm{F}, S_m \rangle$ where $S = S_m \cup S_B \cup S_o$ with

$$S_m = \{(\mathsf{e}, i) \mid \mathsf{e} \in E\} \cup \{\mathsf{e}, \mathsf{e}' \mid \mathsf{e} \# \mathsf{e}'\}$$
$$S_B = \{(Y, \mathsf{e}) \mid Y \mapsto \mathsf{e}\}$$
$$S_o = \{(\mathsf{e}, o) \mid \mathsf{e} \in E\}$$

and F *is defined as follows*

$$\mathrm{F}(s, \mathsf{e}) = \begin{cases} 1 & \textit{if } s = (\mathsf{e}, i) \textit{ or } s = (\mathsf{e}, \mathsf{e}') \textit{ or } s = (\mathsf{e}', \mathsf{e}) \\ 1 & \textit{if } s = (Y, \mathsf{e}) \\ 0 & \textit{otherwise} \end{cases}$$

$$\mathrm{F}(\mathsf{e}, s) = \begin{cases} 1 & \textit{if } (s = (Y, \mathsf{e}') \textit{ and } \mathsf{e} \in Y) \textit{ or } s = (\mathsf{e}, o) \\ 0 & \textit{otherwise} \end{cases}$$

is an unravel net.

Proof. The safeness of $\mathcal{N}(\beta)$ results from the fact that the places (e, i) allows only one execution of each transitions e as they have no incoming arc, and the places with more that one incoming arc, the S_B ones, cannot be marked by more than one transition because of the conflicts in the bundle set, finally the places $\{\mathsf{e}, \mathsf{e}'\}$ does not allow the execution of conflicting transitions in the same run.

Let σ be a fs of $\mathcal{N}(\beta)$ and be $\tau = \mathsf{e}_1 \cdots \mathsf{e}_n = \mathit{run}(\sigma)$ the associated trace. By induction on $\mathit{len}(\tau)$ we show that $N|_{\overline{\mathit{run}(\sigma)}}$ is a conflict-free causal net.

- The empty trace is a conflict-free causal net (a net of marked places only and no transitions),

- consider the trace $\tau = \mathsf{e}_1 \cdots \mathsf{e}_{n-1}\mathsf{e}_n$ and let $\overline{\tau}$ be the state $\{\mathsf{e}_1, \ldots, \mathsf{e}_{n-1}\mathsf{e}_n\}$ associated to τ. As $\tau' = \mathsf{e}_1 \cdots \mathsf{e}_{n-1}$ is a trace as well, by inductive hypothesis

$\mathcal{N}(\beta)|_{\overline{\tau'}} = \langle {}^\bullet\overline{\tau'} \cup \overline{\tau'}^\bullet \cup S_m, \overline{\tau'}, \mathrm{F}', S_m \rangle$ is a conflict-free causal net. Consider $\mathcal{N}(\beta)|_{\overline{\tau}}$ and assume that a place s in $\mathsf{e}_n{}^\bullet$ already belongs to ${}^\bullet\overline{\tau'} \cup \overline{\tau'}^\bullet$. Then s must be a place of the kind (Y, e_i), for some $\mathsf{e}_i \in \overline{\tau'}$, but then there must be a e_j in $\overline{\tau'}$ such that $\mathsf{e}_j \in Y$, which this contradicts the fact that $\overline{\tau'}$ is conflict-free as $\mathsf{e}_j \# \mathsf{e}_n$. So e_n cannot mark places already marked in the past, hence the subnet $\mathcal{N}(\beta)|_{\overline{\tau'}}$ is a conflict-free causal net. ■

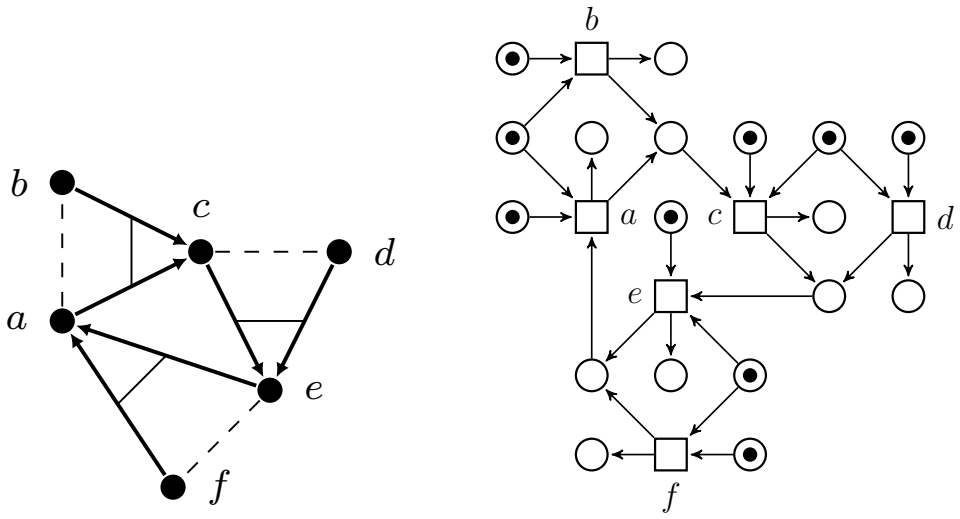
This proposition just establishes that the construction gives an unravel net. In Fig. 3.4 we see an unravel net built from a BES. We show now that the construction is indeed correct as the set of traces of the BES and of the associated net are the same

Theorem 3.2. *Let $\beta = (\mathsf{E}, \mapsto, \#)$ be a* BES *such that $\forall \mathsf{e} \in \mathsf{E}\ \exists X \in \mathsf{Conf}_{fin}(\beta).\ \mathsf{e} \in X$, and $\mathcal{N}(\beta)$ the associated unravel net. Then*

$$\tau \in \mathsf{Tr}(\beta) \iff \tau \in \mathsf{Tr}(\mathcal{N}(\beta))$$

Proof. ($\Rightarrow$) By induction on the length of a trace τ. If τ is the empty trace, then τ is also a trace of $\mathcal{N}(\beta)$. If $len(\tau) = 1$ and $\overline{\tau} = \{\mathsf{e}\}$ then e has no bundle and ${}^\bullet\mathsf{e} \subseteq S_m$, *i.e.* hence e is also a trace of $\mathcal{N}(\beta)$.

Assume that, for $len(\tau) = n$, it holds $\tau \in \mathsf{Tr}(\mathcal{N}(\beta))$, and consider the trace $\tau\mathsf{e}_{n+1} \in \mathsf{Tr}(\beta)$. If e_{n+1} requires no bundle then, by construction, the transition in e_{n+1} is enabled at the marking $lead(\sigma)$ where σ is the fs associated to the trace τ, as the preset of e_{n+1} just contains (e_{n+1}, i) and $\{\mathsf{e}_{n+1}, \mathsf{e}\}$ for each e in conflict with e_{n+1}. But as e_{n+1} has not been executed and τ does not contain any event in conflict with e_{n+1}, they are still marked in $lead(\sigma)$. If it requires bundles, by Def. 3.5, $\overline{\tau} \cap B$ is a singleton $\{\mathsf{e}_i\}$, for some $i \leqslant n$ and for all bundles $B \mapsto \mathsf{e}_{n+1}$. Each e_i can be fired in $\mathcal{N}(\beta)$ by inductive hypothesis and each bundle $B \mapsto \mathsf{e}_{n+1}$) is a place in ${}^\bullet\mathsf{e}_{n+1}$, hence ${}^\bullet\mathsf{e}_{n+1} \subseteq lead(\sigma)$, with σ being the fs associated to τ, is satisfied.



(a) (b)

Figure 3.4: A BES with its associated unravel net.

($\Leftarrow$) Again we do the proof by induction on the length of the trace. For the empty trace the thesis is trivial. Let $\tau = \mathsf{e}_1 \cdots \mathsf{e}_{n-1} \in \mathsf{Tr}(\mathcal{N}(\beta))$. Then there exists a fs σ such that $run(\sigma) = \tau$. Consider the trace $\tau \cdot \mathsf{e}_n \in \mathsf{Tr}(\mathcal{N}(\beta))$. By inductive hypothesis $\mathsf{e}_1 \cdots \mathsf{e}_{n-1}$ is a trace of β, and that e_n is enabled at the marking $lead(()\sigma)$. We show that, for each bundle $B \mapsto \mathsf{e}_n$, $B \cap \{\mathsf{e}_1, \ldots, \mathsf{e}_{n-1}\} \neq \varnothing$. As e_n is enabled at the marking $lead(()\sigma)$, it means that the place (B, e_n) is marked $lead(()\sigma)$, but then a transition $\mathsf{e}_j \in B$ belongs to $\overline{\tau}$, hence $B \cap \{\mathsf{e}_1, \ldots, \mathsf{e}_{n-1}\} \neq \varnothing$. Clearly $\overline{\tau \cdot \mathsf{e}_n}$ is conflict free. ■

Chapter 4

Unravel nets and Unfoldings

In the previous chapter we have introduced unravel nets and we have shown that they are closely related to a particular class of event structures. As bundle event structures model disjunctive causality without duplicating events, they may be the proper kind of event structure for representing the behaviors of a net in a more *compact* way. In this chapter we review the two known techniques for compacting the unfolding of a net and show how they fit into the framework we are trying to develop. The next two sections are dedicated to recall the basic definitions and properties of those approaches, then we discuss whether they do give unravel nets and in the case they don't, if this can be easily enforced.

4.1 Merged processes

Although the existence of a finite representation of an unfolding, the complete prefix, makes it usable in practical applications, the size of the finite prefix can still be too large. In [KKKV06] the authors developed a technique, called *merged processes*, that allows to *compress* the behavior of a Petri net. The idea behind the merged processes is to merge two or more nodes of a branching process that, in some sense, represent the same resource, and then consider as the same the transitions that, after the fusion, share the preset and postset. In this section we

will recall some basic definitions and propositions that we will use in the following chapters.

The idea of being the *same* resource in the case of safe nets is quite simple: the execution of transition t putting a token in a place s just produce the resource in that place for the i-th time, hence to know if two conditions in a branching process represent the same resource it is enough to *determine* the *occurrence-depth* of this condition, namely how many conditions with the same label are less or equal to it. Clearly conditions with the same occurrence-depth and the same label must be in conflict.

Definition 4.1. *Let $b = (C, p)$ be a branching process of a safe net N and x one of its nodes (condition or event). The* occurrence-depth *of x is defined as the maximum number of $p(x)$-labelled nodes on any directed path starting at a minimal (*w.r.t $<$*) condition and terminating at x in the directed graph representing b.*

The above notion is well-defined since there is always at least one directed path starting at a minimal (*w.r.t.* $<$) condition and terminating at x, and the number of all such paths is finite.

Once the occurrence-depth of conflicting conditions has been determined, we can *merge* these conditions and after that merge also the events having the same label, the same preset and the same postset (thus the events representing the same transition in alternative executions).

Definition 4.2. *Let N be a safe net and (C, p) be a branching process of N, where $C = \langle B, E, F, \mathsf{m}\rangle$ is a causal net and p is a labeling mapping satisfying the requirements of Def. 2.13. The merged process of (C, p) is the net $\mathfrak{Merge}(C, p)$ defined by the following steps:*

1. *all the conditions bearing the same label and having the same* occurrence-depth *are* merged *together, and these conditions, called* mp*-conditions, inherits the same incoming and outgoing arcs of the conditions that are fused, finally an* mp*-condition inherits the same label as the fused conditions,*

2. *after performing the previous step, all the events with the same label, the same preset and the same postset are* fused *together, giving an* mp-*event, and they inherit the label from the fused events as well as the incoming and outgoing arcs, and*

3. *the initial marking is given by the* mp-*conditions which are originated by conditions that were minimal in the causal net C.*

If (C, p) is a branching process of a net N and $\mathfrak{Merge}(C, p)$ its merged process, we denote by $\hat{C}$ the net obtained by fusing the conditions and the events and, in correspondence with this operation, adapting the flow relation, the initial marking and the labeling. The latter is well defined as conditions and events are merged only when they bear the same labels. By $\bar{p}$ we denote the labeling (the net morphism $\bar{p} : \hat{E} \to N$) of fused conditions and events on the place and transition of N, and with $\hat{B}, \hat{E}, \hat{F}$ and $\hat{m}$ we denote respectively the set of its mp-events, the set of its mp-conditions, its flow relation and its initial marking. The merged process of the whole unfolding of the net N will be denoted by $\mathfrak{Merge}(N)$.

4.1.1 Properties

We point out some properties of merged processes. Let (C, p) be a merged process of N.

1) There is at most one mp-condition s_k resulting from the fusion of conditions labelled by place s of N occurring at depth $k \geqslant 1$,

2) two distinct conditions in (C, p) having the same label and occurrence-depth are in conflict (it is true for safe Petri nets only, otherwise they can be concurrent),

3) for two mp-conditions, s_k and s_{k+1}, there is a directed path from the former to latter. Moreover, if s_{k+1} is present and $k \geqslant 1$ then s_k is also present,

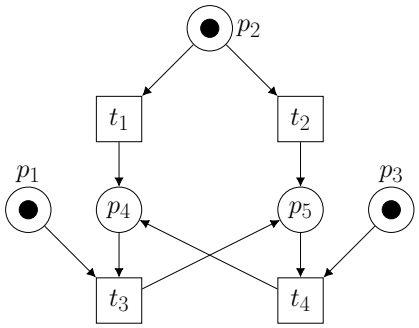


Figure 4.1: The net N running example of [KKKV06]

4) in general, $\hat{C}$ is not acyclic (cycles can arise due to criss-cross fusions of conditions),

5) there can be events consuming conditions in the postset of a cut-off mp-event,

6) there is a strong correspondence between the runs of N and those of $\mathfrak{Merge}(N)$: σ is a run of N iff $\sigma = \hat{h}(\hat{\sigma})$ for some run $\hat{\sigma}$ of $\mathfrak{Merge}(N)$.

Given a merged process $\mathfrak{Merge}(C, p)$, a multiset $\hat{X}$ of mp-events, it is an mp-configuration if $\hat{X} = \bar{p}(X')$ for some configuration X' of $\mathfrak{Merge}(N)$. Note that there is a subtlety in this definition: we have to use the whole unfolding of N rather than an arbitrary branching process (C', p') such that $\mathfrak{Merge}(C, p) = \mathfrak{Merge}(C', p'))$, since $\mathfrak{Merge}(C, p)$ may contain mp-configurations which are not $\bar{p}$-images of any configurations in such a branching process, i.e., the mp-configurations of $\mathfrak{Merge}(C, p)$ might be ill-defined if it can arise from several different branching processes.

Example 4.1. *Consider the net in Fig 4.1, in its finite prefix there are two conditions labeled with p_4, both with occurrence depth equal to 1, and similarly for the two conditions labeled with p_5. Hence these two pairs of conditions can be merged giving the merged process in Fig. 4.2. Observe that this is not an unravel net. In fact the executions involving the two events e, e' labeled with t_3 and t_4 respectively,*

mark twice one of the p_i^1, with $i \in \{4, 5\}$ (depending whether the event labeled with t_1 or the one labeled with t_2 has been executed).

The previous example shows that in general the notion of merged process does not enforce that the resulting net is an unravel one. We will see later how to enforce this property.

4.2 Trellis processes

Instead of the token occurrence, the *time* can be taken into account when merging conflicting conditions of a branching process.

To formalize this idea, on which the notion of *trellis* process is based (see [Fab07]), we do need to guarantee that the proper time can be identified for each condition. To do so we resort to *multi-clock* nets, introduced by E. Fabre ([Fab07]). First of all we observe that if N is a multi-clock net then also any branching process of N is a causal net as well and the partition mapping is induced by the the one for N.

Let $\mathsf{C} = (C, p)$ be a branching process of the multi-clock N, with ν as partition mapping. Given a condition b of C, the *height* of b, denoted with $\mathsf{height}(b)$, is $|\{b' \in B \mid b' \leqslant_C b \text{ and } \nu(p(b)) = \nu(p(b'))\}|$, where B are the conditions of C. The height of a condition is well defined in a casual net which is a multi-clock as well, as in the case of branching processes arising from multi-clock nets.

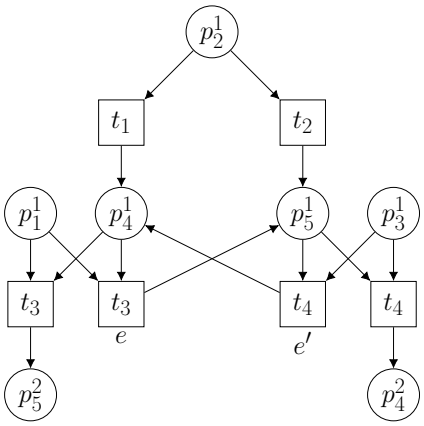


Figure 4.2: Merged process of the net in Fig 4.1

We first recall the notion of *trellis*.

Definition 4.3. *A* trellis $\mathcal{T} = \langle B, E, G, \mathsf{c}, \nu \rangle$ *is a (multi-clock) net satisfying:*

1) $\mathsf{c} = \{b \in B \mid {}^\bullet b = \varnothing\}$,

2) for every $b \in B$*, the automaton* $\mathcal{T}|_{\overline{\nu(b)}}$ *is acyclic (i.e. its flow relation defines a partial order), and*

3) each event belongs to at least one finite state of $\mathcal{T}$*, i.e.* $\forall e \in E.\ \exists X \in \mathsf{St}(\mathcal{T}).\ e \in X$.

The notion of trellis has, like the one of unravel net, an acyclicity requirement, and this is local and also syntactic, as it depends on the partition mapping only. Trellis nets are indeed *unravel net*, as conditions 2 and 3 of the definition ensure that each state is acyclic and each event is at least in a state.

Proposition 4.1. *Let* $\mathcal{T}$ *a trellis net, then* $\mathcal{T}$ *is an unravel net.*

Proof. See Lemma 2 in [Fab07]. ■

We can recall the notion of trellis process.

Definition 4.4. *Let* $N = \langle S, T, F, \mathsf{m}, \nu \rangle$ *be a multi-clock net. Then a* trellis process *of* N *is the pair* $(\mathcal{T}, p)$ *where* $\mathcal{T} = \langle B, E, G, \mathsf{c}', \nu \rangle$ *is a trellis net and* p *a labeling mapping* $p\colon B \cup E \to S \cup T$ *such that*

1) p *is a folding,*

2) $\forall e, e' \in E$ *if* ${}^\bullet e = {}^\bullet e' \wedge p(e) = p(e')$ *then* $e = e'$*, and*

3) $\forall b, b' \in B$ *if* $\mathsf{height}(b) = \mathsf{height}(b') \wedge p(b) = p(b')$ *then* $b = b'$.

Example 4.2. *Consider the multi-clock net in Fig. 4.3(a). The net has three components (one has places* $\{a, b\}$*, another the places* $\{c, g\}$ *and the last one places* $\{d, e, f\}$*. The initial part of the trellis is the one in Fig. 4.3(b). The condition* c

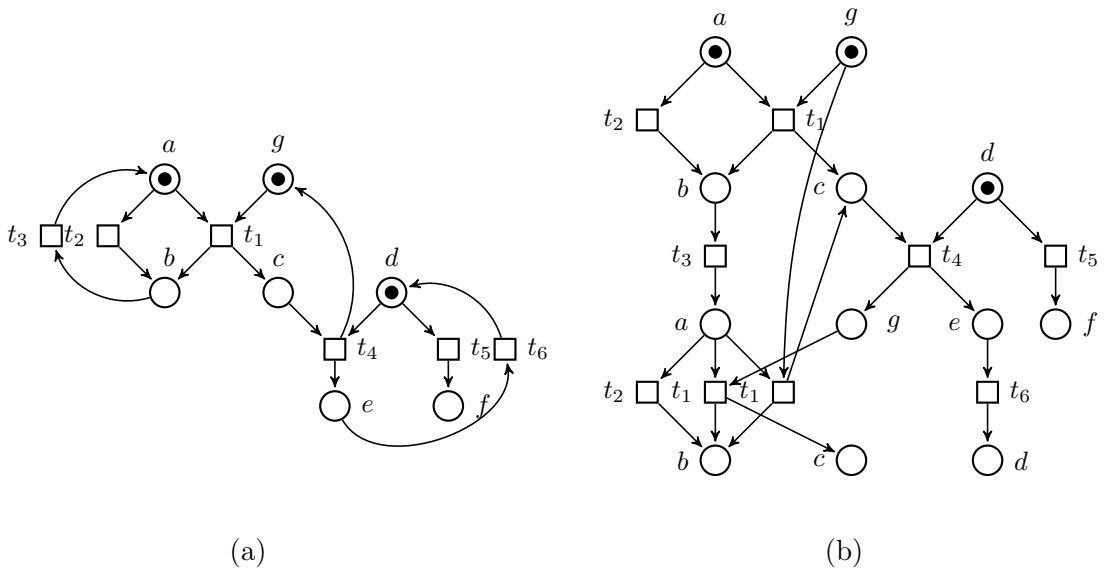


Figure 4.3: A multi-clock net and one of its trellis process.

belonging to the second component is fused as it belongs either to the computatione where the first two automata do synchronize on t_1 or to the one where the first automata does a t_2 followed by t_3 before synchronizing on t_1.

Trellis processes of a multi-clock net N can be obtained by a branching process fusing conditions belonging to the same partition, having the same label and the same height.

4.3 Turning merged processes into unravel nets

Merged processes are not, in general, unravel nets, as we shown in the Ex. 4.1. The main reason is that whenever two or more transitions are marked as equivalent, it may happen that conflicts in the past are forgotten, allowing executions that would be otherwise forbidden.

For instance, consider the net in Fig. 4.4. The branching process which is the prefix is the one in Fig. 4.5 and the corresponding merged process is the one in Fig. 4.6. This is not an unravel net. In fact, consider the runs *abcd* and *bcd*. After the merging, the postset of q_1 ($q_1{}^\bullet$) contains $d, \bar{d}$, which have the same label

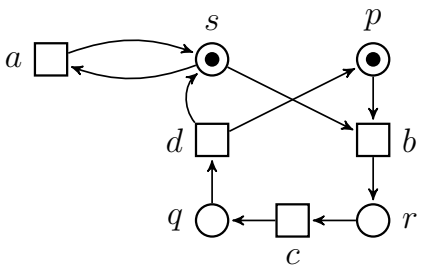


Figure 4.4: A safe net N

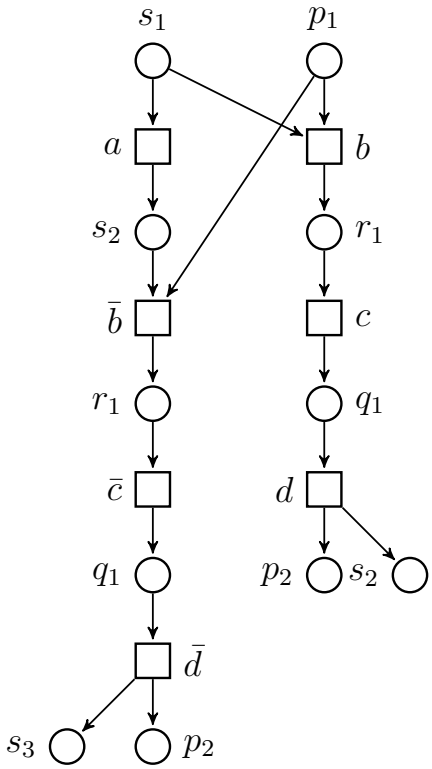


Figure 4.5: The prefix (C, p) of the net N in Fig. 4.4.

(thus they represent the happening of the same transitions of the safe net N in two different context). It is clear that only one of them should be executable, depending on the path we followed to mark q_1 because $d \# a$ but $\neg(\bar{d}\#a)$. However this can be turned into an unravel net by suitably adding some places to rule out unwanted executions. This has to be done with some criteria and in the following we propose a criterion.

4.3.1 Conflict conditions

The idea is rather simple: for each place in the original net N, we add a condition in the representation of the behaviors of the net N with the aim of representing that the $i-th$ token has been produced that place. Then the idea is to make sure that all the events producing the $i-th$ token in that place (which is a condition or a set of conditions) produce this condition and *use* the conditions stating that the $(i-1)-th$ token was produced before in that place. This intuition is formalized as follows:

Definition 4.5. *Let N be a safe net, (C,p) a branching process of N, and $(\hat{C},\hat{p})$ be the merged process of (C,p), where $\hat{C} = \langle \hat{B},\hat{E},\hat{G},\hat{\mathsf{c}}\rangle$. We define* $\mathsf{Ng}(\hat{C}) =$

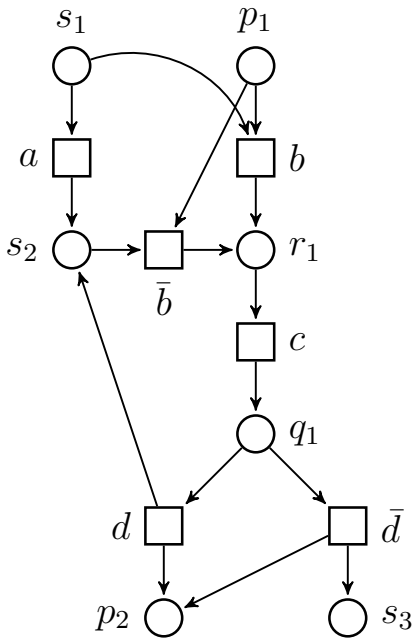


Figure 4.6: The merged process of the net N in Fig. 4.4 corresponding to the branching process in Fig. 4.5.

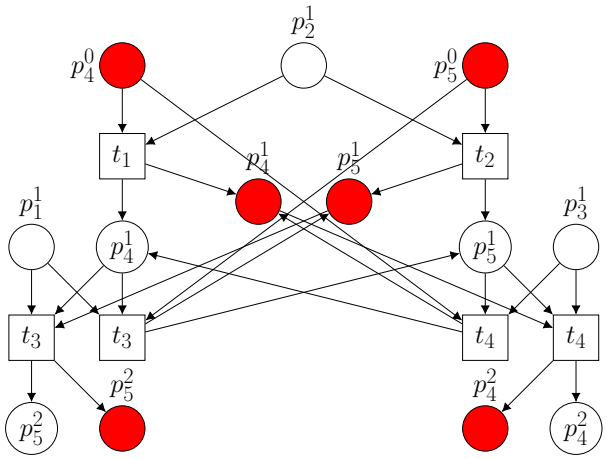


Figure 4.7: The enriched merged process of Fig 4.1

$\langle \hat{B} \cup B_{\#}, \hat{E}, \overline{G}, \overline{\mathsf{c}}, \overline{p} \rangle$ *as the net where:*

$$B_{\#} = \{(b, \#, i) \mid b \in B \wedge \textit{ the occurrence depth of } b \textit{ is } i\} \cup \{(b, \#, 0) \mid b \notin \mathsf{c}\}$$

$$\overline{G}(x,y) = \begin{cases} \hat{G}(x,y) & \textit{if } \{x,y\} \subseteq \hat{B} \cup \hat{E} \\ 1 & \textit{if } x = (b,\#,i),\ b \in y^\bullet \wedge \textit{ the occurrence depth of } b \textit{ is } i+1 \\ 1 & \textit{if } y = (b,\#,i),\ b \in x^\bullet \wedge \textit{ the occurrence depth of } b \textit{ is } i \end{cases}$$

$$\overline{\mathsf{m}}(x) = \begin{cases} 1 & \textit{if } x = (b,\#,0) \\ 1 & \textit{if } x = b \textit{ and } b \in \hat{\mathsf{c}} \\ 0 & \textit{otherwise} \end{cases}$$

The conditions in $B_{\#}$ *are called* no-gap *or* conflict *conditions.*

Example 4.3. *Consider the merged process of the Ex. 4.1. The result of enriching the net in Fig. 4.1 is shown in Fig. 4.7, where conflict conditions are depicted as circles filled in red.*

The result of this construction is an unravel net.

Theorem 4.1. *Let* N *be a safe net,* (C, p) *a branching process of* N*, and* $\mathfrak{Merge}(C, p)$ *be the corresponding merged process. Then* $\mathsf{Ng}(\mathfrak{Merge}(C, p))$ *is an unravel net.*

Proof. We first show that this is indeed a safe net, and then we prove that each execution gives an acyclic net.

- We first show that $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ is a safe net. To this aim we recall that the merged process of a safe net is a safe net (see [KKKV06]). By adding conflict conditions between two events we syntactically states that they are images of two conflict events of the branching process. To show this consider two transition e and e' of (C,p) such that $\neg(e\# e')$, with $\{b\} \subseteq t^\bullet \cap t'^\bullet$, and assume that the occurrence-depth of the conflicting conditions that are fused in b is k, and assume that, after the enrichment with conflict conditions, they are still not conflicting, *i.e.* there is a state X of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ such that $\{e,e'\} \subseteq X$.

  If k is one, then both events use the same initially marked condition $(b,\#,0)$, then $e\#e'$ are in conflict, contradicting the assumption.

  Suppose $k \geqslant 2$. An event of X putting a token in $(b,\#,k-1)$ needs a token from $(b,\#,k-2)$ and so on, thus a state X of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ in which $(b,\#,k)$ is marked will contain all conflict conditions $(b,\#,i)$, $0 \leqslant i < k$. The only way to mark $(b,\#,k)$ in the same state is to enable the preset of $(b,\#,k-1)$ two times (at least). By repeating the reasoning till $(b,\#,1)$, using the fact that ${}^\bullet(b,\#,1)$ is composed by a set of conflicting events, and the impossibility to mark $(b,\#,0)$ again in a state, it follows that if e is enabled in a state then e' cannot be enabled as well. Hence the enriched net is safe.

- In the previous item we have shown the impossibility to obtain an executable loop in the merged process enriched with conflict condition (it would violate the safeness). We also showed that two (or more) events putting a token in the same (resource or conflict) place are in conflict. To summarize states of this net are sets of events (the net $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ is an occurrence net),

and no place is marked again (the net $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ is acyclic). Hence the thesis.

■

We now prove that states of this unravel net are precisely those we are interested in.

Theorem 4.2. *Let N a safe net, (C,p) a branching process of N, $\mathfrak{Merge}(C,p)$ be the merged process of (C,p) and $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ the corresponding enriched net. Then X is a configuration of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ iff σ is a mp-configuration of $\mathfrak{Merge}(C,p)$.*

Proof. ($\Rightarrow$)

We must prove that:

1. $X \in \mathsf{St}(\mathsf{Ng}(\mathfrak{Merge}(C,p))) \;\Rightarrow\; X \in \mathsf{St}(\mathfrak{Merge}(C,p))$,
2. $\mathsf{Ng}(\mathfrak{Merge}(C,p))|_X$ is acyclic,
3. denote with B_i the conditions of $\mathsf{Ng}(\mathfrak{Merge}(C,p))|_X$ and consider $b \in B_i$ such that b is the merged condition corresponding to conditions b' of (C,p) such that the occurrence depth of b' is k, then all conditions $\hat{b}$ corresponding to conditions b'' of (C,p), bearing the same label as b, such that the occurrence depth of b' is i, $0 \leqslant i < k$, are present in B_i as well,

Removing places (conditions) from a Petri net can only increase the number of firing sequences, so (1) is trivially true. To prove (2) it should be noted that, since the events of X identify an acyclic subnet in the enriched net $\mathsf{Ng}(\mathfrak{Merge}(C,p))$, that subnet still enjoys the property of being acyclic if we remove a subset of conditions (the no-gap one).

If a condition $b \in B_i$ corresponding to conditions with token occurrence k (b_k) is present in X, this means the event e such that $b_k \in e^\bullet$, consume the token

$(b, \#, k-1)$ marked by e' with $b_{k-1} \in e'^{\bullet}$. Since $\mathsf{Ng}(\mathfrak{Merge}(C,p))|_X$ is acyclic, we can proceed backward till we encounter b_0, if the place $p(b) \notin \mathsf{m}$ or p_1 otherwise, hence the thesis.

($\Leftarrow$)

Let X be a mp-configuration of $\mathfrak{Merge}(C,p)$. We show that it is a configuration of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$. (by induction on the size of the mp conf?)

Assume $X = \{e\}$. The preset of e in $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ contains only marked conditions (no-gap and resources) so its is firable in $\mathsf{Ng}(\mathfrak{Merge}(C,p))$, hence $\{t\}$ is a configuration of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ as well.

Assume that a mp-configuration X, with size $n-1$, is a also a configuration of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$. Consider the extended mp-configuration $X' = X \cup \{e\}$. Let $e^{\bullet}$ be composed by conditions with token occurrence $\mathsf{tok}_1, \ldots, \mathsf{tok}_k$. By definition of mp-configuration there is a set of events in X which marks a set of conditions that are the fusion of conditions with token occurrence $\mathsf{tok}_1 - 1, \ldots, \mathsf{tok}_k - 1$ (if $\mathsf{tok}_k - 1 = 0$ then that condition is initially marked). After the execution of X in $\mathsf{Ng}(\mathfrak{Merge}(C,p))$ the conflict condition $(b, \#, \mathsf{tok}_1 - 1), \ldots, (b, \#, \mathsf{tok}_k - 1)$ are marked. Assume that one of them is not marked, then there would be an event $\bar{e} \neq e'$ which consumed that token and has marked the condition $\bar{b}$ corresponding to conditions with occurrence-depth tok_i, $1 \leqslant i \leqslant k$. But this would violate the safeness of the mp-configuration since e' marks $\bar{b}$ too. To summarize X marks all the conflict condition of e' and thus X' is also a configuration of $\mathsf{Ng}(\mathfrak{Merge}(C,p))$.

■

4.4 Conflict conditions and trellises

Trellis processes executions are acyclic, then it may sound useless to enrich them with the conflict conditions. But, since we aim to use unravel nets to build a general theory for compacting net behaviors, it may be useful to show how to add conflict conditions in a setting where token occurrence does not play a role.

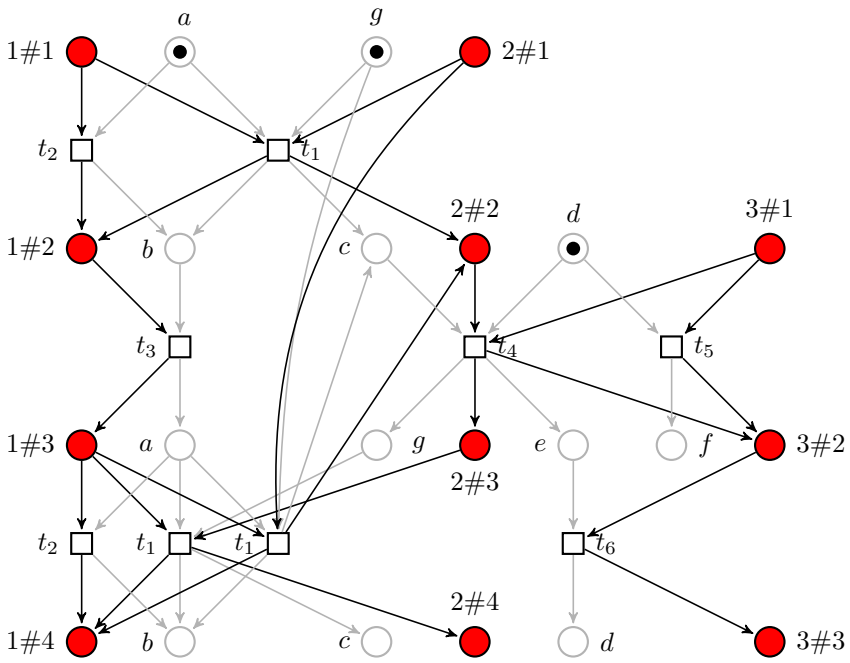


Figure 4.8: The enriched trellis obtained from the one in Fig. 4.3b

From the merged process point of view a conflict condition $(b, \#, k)$ is added to the preset of an event e if it put a token in $b \in B_i$ with occurrence depth of b equal to k, an arc from $(b, \#, k-1)$ to e is also set. For trellises we have to take into account the height of each condition, which depends on the automaton it belongs to. The set of conflict conditions $B_\#$ is then defined as follows (and has no relation with the order in which some resources are produced but with the *local time* of production):

$$B_\# = \{(\nu(b), \#, i) \mid b \in B \wedge \mathsf{height}(b) = i\}$$

The flow relation connecting these new conditions to events is defined in the obvious way, and it is also trivial to establish which ones are the initial conditions. The whole net is not any longer a multi clock net as now conflict conditions belonging to a partition are together with the conditions of this partition, thus violating one of the requirements for being a multi clock net. Still, if we restrict to resource conditions or conflict conditions we obtain again a multi clock net.

Example 4.4. *In Fig 4.8 we see how to enrich a trellis process in Fig. 4.3b of the multi-clock net in Fig. 4.3a. It should be stressed that, since trellis processes are already unravel nets, conflict places are somehow superfluous, but they stress which events belonging to the same component are in conflict.*

Chapter 5

Merging relations

In this chapter we propose a simple and general framework to compact labeled unravel nets. We assume that labeled unravel net represent the behavior of the system (in general a net), and the labeling mapping is the one relating the places and transitions of the unravel net to the one of the system.

We introduce the notion of *incompatibility* among places which can be used to identify the places that can be *fused*. Then we define the notion of merging relation which can be used to compact Petri nets where incompatible places have been identified. We show that this framework includes the notion of merged processes and trellis processes, as we will show that the criteria for identifying places are indeed merging relations. As it should be clear, merging an unravel net does not necessarily gives an unravel net. We investigate under which conditions a merging relation gives an unravel net from another one, or similarly to what we have done with merging process, how to enrich the merged unravel net to guarantee that the result is still an unravel net.

5.1 Merging contexts

We first introduce a *semantic* notion of incompatibility between places, capturing the idea that, if a place is akin to a *resource*, two resources are incompatible if

they never appear in the same computation, even at different stages. Then we show that, given a suitable equivalence relation, related to places incompatibility, a more succinct version of the net we started with can be obtained.

Let $\mathsf{N} = (N, l)$ be a labeled net where $N = \langle S, T, F, \mathsf{m}\rangle$ and let $t, t' \in T$ be two transitions. We say that t and t' are *identifiable* whenever ${}^\bullet t = {}^\bullet t'$, $t^\bullet = t'^\bullet$ and $l(t) = l(t')$. Thus on T it is possible to define an equivalence relation $\simeq$ such that $t \simeq t'$ iff t and t' are identifiable. The set of transitions can be quotiented through this equivalence relation obtaining the set $\{[t]_\simeq \mid t \in T\}$. Observe that the transitions in $[t]_\simeq$ are such that they cannot be executed *together* at the same marking.

Definition 5.1. *Let $(N, l) = (\langle S, T, F, \mathsf{m}\rangle, l)$ be a labeled net and let $\simeq$ be the equivalence relation induced by transitions identifiability. Then we can construct the labeled net $\widehat{\mathsf{N}} = (\widehat{N}, \hat{l})$ where $\widehat{N}$ is the Petri net $\langle S, \widehat{T}, \widehat{F}, \mathsf{m}\rangle$ with $\widehat{T} = \{[t]_\simeq \mid t \in T\}$, $\widehat{F}(s, [t]_\simeq) = F(s, t)$ and $\widehat{F}([t]_\simeq, s) = F(t, s)$, and $\widehat{l}$ is the mapping defined as $\hat{l}(s) = l(s)$ and $\hat{l}([t]_\simeq) = l(t)$.*

Let $\sigma \in \mathcal{R}^N_\mathsf{m}$, then $\widehat{\sigma}$ is constructed as follows: $\widehat{\sigma} = \mathsf{m}$ if $\sigma = m$ and $\widehat{\sigma} = \widehat{\sigma'}\,[[t]_\simeq\rangle m$ if $\sigma = \sigma'\,[t\rangle m$. The firing sequences of N and of $\widehat{N}$ are clearly related, as the following proposition shows. Observe that $run(\widehat{\sigma}) = l([t_1]_\simeq \cdots [t_n]_\simeq) = run(\sigma)$.

Lemma 5.1. *Let $(N, l) = (\langle S, T, F, \mathsf{m}\rangle, l)$ be a labeled net, and let $\simeq$ be the equivalence relation induced by transitions identifiability. Let $\widehat{\mathsf{N}} = (\widehat{N}, \hat{l})$ as in Def. 5.1. Then $\langle h_S, h_T\rangle \colon N \to \widehat{N}$ defined as $h_T(t) = [t]_\simeq$, $h_S = i_S$ is a net morphism.*

Proof. Since h_S is the identity relation, each place of the initial marking of N is related to itself. By definition each transition belong to an equivalence class, so h_T is total, in particular also its restriction to ${}^\bullet s$ and $s^\bullet$, for each s, is total as well (since $s\ h_S\ s$). As h_S is the identity relation, h^{op}_s, which is the identity function, is total. Then, by Def. 2.14, $\langle h_S, h_T\rangle$ is a net morphism. ∎

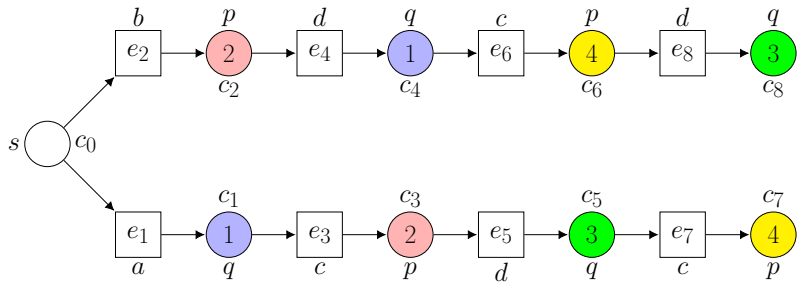


Figure 5.1: A labeled unravel net N.

Proposition 5.1. *Let $(N, l) = (\langle S, T, F, \mathsf{m}\rangle, l)$ be a labeled net, and let $\simeq$ be the equivalence relation induced by transitions identifiability. Let $\sigma \in \mathcal{R}^N_{\mathsf{m}}$ be a fs, then there exists a fs $\sigma' \in \mathcal{R}^{\widehat{N}}_{\mathsf{m}}$ such that $\widehat{\sigma} = \sigma'$.*

Proof. Follows from Prop. 5.1 and Th. 2.1 as firing sequences are preserved. ■

5.1.1 Places Incompatibility

Here we introduce a notion of place incompatibility (*i.e.* resources of the safe net). The notion is semantical as it depends on all the firing sequences.

Definition 5.2. *Let $N = \langle S, T, F, \mathsf{m}\rangle$, we say that two places $s, s' \in S$ are* incompatible *iff for each firing sequence $\sigma \in \mathcal{R}^N_m$ it holds that $\{s, s'\} \nsubseteq \mathsf{P}(\mathsf{M}(\sigma))$, and we denote it with $s \bowtie s'$*

Observe that this notion is quite similar to the one of conflict we introduced on unravel nets.

Clearly $\bowtie$ is a symmetric relation. Observe that if $s \in [\![\mathsf{m}]\!]$ and $s \bowtie s'$ then $\forall \sigma \in \mathcal{R}_m.\ s' \notin \mathsf{P}(\mathsf{M}(\sigma))$. We are interested in a *conflict* relation on places that implies incompatibility, but does not necessarily coincide with it.

Example 5.1. *Consider the labeled net $\mathsf{N} = (N, l)$ in Fig. 5.1, with initial marking $\mathsf{m} = \{c_0\}$. The relation $\bowtie$ contains the pairs (c_i, c_j) such that $i, j > 0$ and if i is odd then j is even and vice versa as well. Thus $c_4 \bowtie c_7$ and $c_7 \bowtie c_6$ but $c_6 \bowtie\!\!\!\!\!\!/\ \ c_4$.*

5.1.2 Merging relation

We introduce now the main notion of the chapter, namely the one of *merging relation*. A merging relation is any equivalence relation induced by another arbitrary relation which is included in the *incompatible* one.

Definition 5.3. *Let* $\mathsf{N} = (N, l)$ *be a labeled net where* $N = \langle S, T, F, \mathsf{m}\rangle$*, let* $\circledast \subseteq \bowtie$ *be a transitive relation, and let* $\sim$ *be an equivalent relation such that* $s \sim s' \Leftrightarrow (s \circledast s' \vee s = s') \wedge l(s) = l(s')$*. Then* $\sim$ *is a* merging relation *for* N*.*

Thus a merging relation is any equivalence relation respecting labeling and incompatibility. Observe that the identity on places is a trivial merging relation. Furthermore if s is initially marked then $[s]_\sim = \{s\}$.

The *merging* relation is used to *compress* the net. Similarly to what is done in [KKKV06], we first merge places by identifying equivalent ones, thus the merged places will be $S' = \{[s]_\sim \mid s \in S\}$. Then, when needed, we may identify also transitions.

Definition 5.4. *Let* $\mathsf{N} = (N, l)$ *be a labeled unravel net where* $N = \langle S, T, F, \mathsf{m}\rangle$*, and let* $\sim$ *be a merging relation. Then we construct the labeled net* $\widetilde{\mathsf{N}} = (\widetilde{N}, \widetilde{l})$*, where* $\widetilde{N}$ *is the Petri net* $\langle \widetilde{S}, T, \widetilde{F}, \widetilde{\mathsf{m}}\rangle$ *defined as* $\widetilde{S} = \{[s]_\sim \mid s \in S\}$*,* $\widetilde{F}([s]_\sim, t) = F(s, t)$*,* $\widetilde{F}(t, [s]_\sim) = F(t, s)$ *and* $\widetilde{\mathsf{m}}([s]_\sim) = \sum_{s \in [s]_\sim} \mathsf{m}(s)$*, and* $\widetilde{l}$ *is the labeling mapping defined as* $\widetilde{l}([s]_\sim) = l(s)$ *and* $\widetilde{l}(t) = l(t)$*.*

The flow relation is well defined, as $\forall t \in T.\ |[\![{}^\bullet t]\!] \cap [s]_\sim| \leqslant 1$ and $\forall t \in T.\ |[\![t^\bullet]\!] \cap [s]_\sim| \leqslant 1$ as well, and the same for the initial marking, as the equivalence class of each place in the initial marking contains just that place.

Example 5.2. *Consider the net of the Ex. 5.1. A suitable merging relation can be* $c_1 \sim c_4$*,* $c_2 \sim c_3$*,* $c_6 \sim c_7$ *and* $c_5 \sim c_8$ *and the result of the merging of these places is the net in Fig. 5.2. Another merging relations could be* $c_1 \sim' c_8$*,* $c_2 \sim' c_3$*,* $c_6 \sim' c_7$ *and* $c_5 \sim' c_4$*, or simply* $c_1 \sim'' c_4$*. Clearly the resulting more compact net is different from the one depicted before.*

5.1.3 Preserving behaviours

The construction can be lifted to the reachable markings and firing sequence. Let $m \in \mathcal{M}_N$, then $\widetilde{m} \in \mu\widetilde{S}$ is the mapping from $\widetilde{S}$ to $\mathbb{N}$ defined as $\widetilde{m}([s]_\sim) = \sum_{s \in [s]_\sim} m(s)$. Observe that, as places in $[s]_\sim$ are in conflict, hence incompatible, at most one may contain a token. We then lift the construction to firing sequences, and we use the same "$\widetilde{\cdot}$" to denote it. Consider $\sigma \in \mathcal{R}^N_{\mathsf{m}}$, then $\widetilde{\sigma}$ is obtained as follows: if $\sigma = \mathsf{m}$ then $\widetilde{\sigma} = \mathsf{m}$, if $\sigma = \sigma' [t\rangle m$ then $\widetilde{\sigma} = \widetilde{\sigma'} [t\rangle \widetilde{m}$, *i.e.* the new firing sequence comprises the same transitions, which still are enabled at each marking.

The following proposition points out the obvious relation among the firing sequences of both nets.

Proposition 5.2. *Let $\mathsf{N} = (N, l)$ be a labeled unravel net where $N = \langle S, T, F, \mathsf{m}\rangle$, and let $\sim$ be a merging relation. Let $h_T \colon T \to T$ be the identity and $h_S \subseteq S \times \widetilde{S}$ be the relation defined as follows: $s \; h_S \; [s]_\sim$. Then $\langle h_T, h_S\rangle \colon \mathsf{N} \to \widetilde{\mathsf{N}}$ is a net morphism.*

Proof. First of all we observe that if $s \in \mathsf{m}$ we have that $\forall s' \in S.\ s' \neq s. \Rightarrow \neg(s \bowtie s')$. Clearly if $s \in \mathsf{m}$ then $[s]_\sim$ is just $\{s\}$, hence the thesis.

Consider now $s \; h_S \; [s]_\sim$, $h_T \colon {}^\bullet s \to {}^\bullet [s]_\sim$ is the identity, hence it is total.

Finally take $t \in T$ then $h_S^{op} \colon \{[s]_\sim \in \widetilde{S} \mid s \in {}^\bullet t\} \to {}^\bullet t$ is total (it maps $[s]_\sim$ to

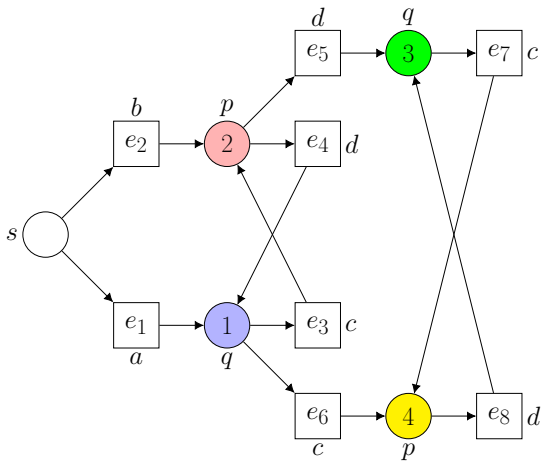


Figure 5.2: The compact representation of the net N in Fig. 5.1

the s in ${}^\bullet t$). Similarly $h^{op}_S \colon \{[s]_\sim \in \widetilde{S} \mid s \in t^\bullet\} \to t^\bullet$ is total. ■

Proposition 5.3. *Let* $\mathsf{N} = (N, l)$ *be a labeled net, let* $\sim$ *be a merging relation and* $(\widetilde{N}, \widetilde{l})$ *be the labeled net of Def. 5.4. Then* $\forall \sigma \in \mathcal{R}^N_{\mathsf{m}}\ \exists \sigma' \in \mathcal{R}^{\widetilde{N}}_{\widetilde{\mathsf{m}}}.\ \widetilde{\sigma} = \sigma'$.

Proof. An obvious consequence of Prop. 5.2, Th. 2.1 and Cor. 2.1. ■

By merging places it may happen that two equally labeled transitions have the same preset and the same postset. Hence the equivalence relation $\simeq$ induced by transition identifiability may be non trivial, *i.e.* different from the identity. We can then apply the construction of Def. 5.1 and identify these transitions as well. The executions of the original net and the ones of the compact version are related as follows.

Proposition 5.4. *Let* $\mathsf{N} = (N, l)$ *be a labeled net and let* $\sim$ *be a merging relation. Then* $(\overline{N}, \overline{l})$ *is the labeled net obtained applying first the construction of Def. 5.4 and then the one of Def. 5.1, and,* $\forall \sigma \in \mathcal{R}^N_{\mathsf{m}}\ \exists \sigma' \in \mathcal{R}^{\overline{N}}_{\overline{\mathsf{m}}}.\ run(\sigma) = run(\sigma')$.

Proof. Before identifying transitions we have that to each $\mathsf{fs}\ \sigma$ of N a $\mathsf{fs}\ \widetilde{\sigma}$ in $\widetilde{N}$ corresponds. Clearly $run(\sigma) = run(\widetilde{\sigma})$. Observing that the identified transitions have the same preset, the same postset and have the same label, we have the thesis. ■

We stress again that by identifying places and transitions we do not lose any behavior, but the obtained net may have more behaviors than the one we started with.

5.2 Merging causal nets

In chapter 4 we recalled two methods for compacting Petri nets, *merged processes* and *trellis processes*. Both start from a causal net as the representation of a net behavior (a branching process) and then merge places and transitions according

to two different criteria. Here we show a quite obvious result: each of two criteria defined on causal nets clearly induces a merging relation.

Merged processes. The merging criteria is the occurrence depth of a transition. Let $\mathsf{C} = (C, p)$ be a branching process of the safe net N, we define the token occurrence of a condition as $\mathsf{tok}(b) = |\{b' \in B \mid b' \leqslant_C b \ \wedge \ p(b) - p(b')\}$. We define a relation $\sim_\mu$ between two conditions $b, b' \in B$ such that:

$$b \sim_\mu b' \Leftrightarrow \mathsf{tok}(b) = \mathsf{tok}(b') \wedge p(b) = p(b')$$

The fact that $b \bowtie b'$ is implied by the fact that C is a causal net, hence two conditions with the same token occurrence should belong to two *alternative* branches of C, and the conflict relation chosen is precisely $\bowtie$. The following proposition states that $\sim_\mu$ is indeed a merging relation.

Proposition 5.5. *$\sim_\mu$ is a merging relation.*

Proof. By property (2) of merged processes (see subsection 4.1.1), two distinct conditions with the same token occurrence are in conflict and have the same label, namely $\forall b, b'.\mathsf{tok}(b) = \mathsf{tok}(b') \wedge p(b) = p(b') \Rightarrow b \ \# \ b'$. The conflict relation on conditions in a causal nets implies incompatibility, *i.e.* $\# \subseteq \bowtie$. Let us prove that $\sim_\mu$ is an equivalence relation.

- *reflexivity*

  $\forall b, \mathsf{tok}(b) = \mathsf{tok}(b) \wedge p(b) = p(b)$, then $b \sim_\mu b'$,

- *symmetry*

  $\forall b, b'$ such that $b \sim_\mu b'$,$\mathsf{tok}(b) = \mathsf{tok}(b') \wedge p(b) = p(b')$ and $=$ is symmetric, hence $b' \sim_\mu b$,

- *transitivity*

  $\forall b, b', b''$ such that $b \sim_\mu b' \wedge b' \sim_\mu b''$, $\mathsf{tok}(b) = \mathsf{tok}(b') \wedge p(b) = p(b')$ and

$\mathsf{tok}(b') = \mathsf{tok}(b'') \wedge p(b') = p(b'')$. From the transitivity of the $=$ relation follows the thesis.

To summarize $b \sim_\mu b' \Leftrightarrow (b \# b' \vee \ b = b') \ \wedge \ p(b) = p(b')$. ■

Trellis processes. For trellises the relation between two conditions of the branching process is defined as:

$$b \sim_\mathcal{T} b' \Leftrightarrow \mathsf{height}(b) = \mathsf{height}(b') \wedge p(b) = p(b')$$

As before, the following proposition holds:

Proposition 5.6. *$\sim_\mathcal{T}$ is a merging relation.*

Proof. The proof is the same of Prop. 5.5. ■

We can conclude this section observing that the two criteria introduced for trellis and merged processes give two merging relations.

5.3 Merging unravel nets

The framework we have devised in the previous sections can be applied to any kind of labeled net representing the behavior of a Petri net. In particular it may be applied to *unravel nets* where causal dependencies can still be dug out, as well as a conflict relation. Indeed, the only requirement we pose on the net representing the behavior of the net is that a *conflict relation* on places can be identified, but still we may imagine that each state of the net corresponds to a reachable marking of the net whose behavior is represented by the unravel net.

When compacting behaviors we start from a labeled unravel net, but the produced net is not necessarily an unravel one (as we have seen in the previous chapter when dealing with merged processes), and in the compaction we may lose the tight correspondence among nets and event structures. In fact cycles may be introduced

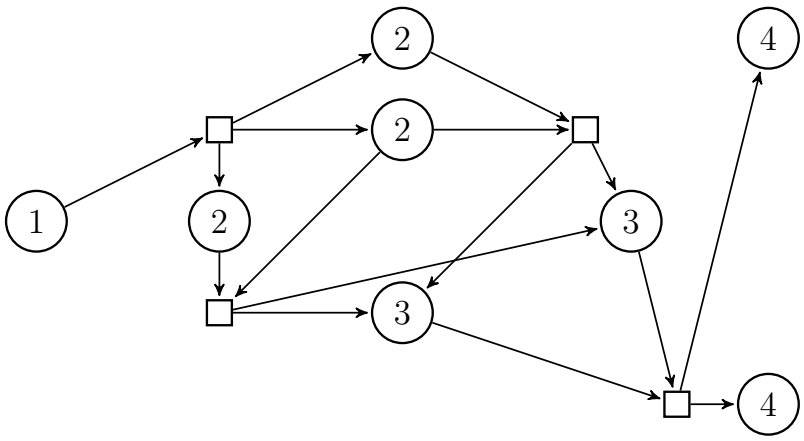


Figure 5.3: Unravel net in which a strictly increasing measure has been defined.

when merging places and transitions, and in some case the cycles may be executable, and thus the associated event structure would have a configuration where dependencies are not a partial order, or even some places may be marked twice (see Ex. 4.1), contrasting the intuition that resources are produced just once (at least in a computation).

We follow closely the approaches in [Fab07, KKKV06] and we try to introduce abstractedly a *measure* on places, on which we base the merging relation. The possibility of preserving the property of being an unravel net after the merging may be then connected with the notion of *measure* associated to places.

Definition 5.5. *Let* $\mathsf{R} = (R, l)$ *with* $R = \langle S, T, F, \mathsf{m} \rangle$ *be a labeled unravel net, and let* δ *be a function such* $\delta : S \to \mathbb{N}$ *such that* $\delta(s) = \delta(s')$ *implies* $s \bowtie s'$*. We say that* δ *is* strictly increasing measure *if for each state* $X \in \mathsf{St}(R)$*, and each pair of places* s, s' *of the net* $R|_{[\![X]\!]}$*, it holds that* $s \leqslant s' \;\Rightarrow\; \delta(s) < \delta(s')$*.*

We state some properties of unravel nets equipped with a strictly increasing measure.

Proposition 5.7. *Let* $R = \langle S, T, F, \mathsf{m}, l \rangle$ *be a labeled unravel net, and let* $\delta : S \to \mathbb{N}$ *be a strictly increasing measure, then* $\forall s, s' \in S$ *such that* $s^\bullet \cap {}^\bullet s' \neq \varnothing$ *it holds* $\delta(s) < \delta(s')$

Proof. Let $t \in s^\bullet \cap {}^\bullet s'$. In any subnet $R|_X$ that contains t (such X exists because each transition is executable in an unravel net) then $\{s, s'\}$ are places of $R|_X$, $s \leqslant s'$ and then $\delta(s) < \delta(s')$ as the measure δ is strictly increasing. ■

Proposition 5.8. *Let $R = \langle S, T, F, \mathsf{m}, l\rangle$ be a labeled unravel net, and let $s_1 s_2 \cdots s_n$ be a sequence of places with $\{s_1, s_2, \ldots, s_n\} \subseteq S$ be such that $\forall i < n, s_i{}^\bullet \cap {}^\bullet s_{i+1} \neq \varnothing$,* i.e. *there is a path from s_1 to s_n. Let $\delta : S \to \mathbb{N}$ be a strictly increasing measure, then $\delta(s_1) < \delta(s_n)$.*

Proof. As for each i such that $1 \leqslant i < n$ we have that $\delta(s_i) < \delta(s_{i+1})$ we have that $\delta(s_1) < \delta(s_n)$. ■

Note that Prop. 5.8 does not imply that there exists a state which define a subnet containing $\{s_1, s_2, \ldots, s_n\}$. The following proposition says if the measure is strictly increasing then the net is acyclic.

Proposition 5.9. *Let $R = \langle S, T, F, \mathsf{m}, l\rangle$ be a labeled unravel net, and let $\delta : S \to \mathbb{N}$ be a strictly increasing measure, then R is acyclic.*

Proof. Assume there is a cycle $s_1 s_2 \cdots s_n s_1$. By Prop.5.8 we would have that $\delta(s_1) < \delta(s_1)$, leading to a contradiction. ■

The following theorem states that the property of being an acyclic unravel net is preserved when a strictly increasing measure is considered.

Theorem 5.1. *Let $\mathsf{R} = (R, l)$ where $R = \langle S, T, F, \mathsf{m}\rangle$ be a labeled unravel net, let $\delta : S \to \mathbb{N}$ be* strictly increasing *and let $\sim$ be the equivalence relation induced by δ,* i.e. *$s \sim s'$ iff $(\delta(s) = \delta(s')$ and $l(s) = l(s'))$ or $s = s'$. Then the resulting compact labeled net $\overline{\mathsf{R}} = (\overline{R}, \bar{l})$ is a labeled unravel net.*

Proof. We prove something stronger, namely that $\overline{\mathsf{R}}$ is acyclic. Suppose there exists a $\mathsf{fs}\ \sigma = m\,[t_0\rangle\, m_1\,[t_1\rangle \cdots [t_{i-1}\rangle\, m_i\,[t_i\rangle \cdots [t_{j-1}\rangle\, m_j\,[t_j\rangle \in \mathcal{R}_m^{\overline{R}}$ such that $m_i \cap m_j \neq \varnothing$ and $\exists [s]_\sim \in m_i \cap m_j$ such that $m_k([s]_\sim) = 0$ for $i < k < j$, *i.e.* $[s]_\sim$ is marked twice in σ.

Consider a $s' \in [s]_\sim$, $\forall [s_{j-1}]_\sim \in m_{j-1}$ it holds that $\delta(s'_{j-1}) < \delta(s'), s'_{j-1} \in [s_{j-1}]_\sim$ because they share a transition (see Prop. 5.7). By proceeding backward till m_i we define a chain of places of R, $s' \cdots s'$ where $\delta(s') < \cdots < \delta(s')$, which is impossible. Since $\delta(s'_{j-1}) < \delta(s'), s'_{j-1} \in [s_{j-1}]_\sim$ holds for each representant of $[s]_\sim$ then there can be no path from $[s]_\sim$ to itself, hence the thesis. ■

The requirement of being an acyclic unravel net is too strong to ask: cyclic unravel nets would be impossible to merge if we allow strictly increasing measures only, if we want to preserve the property of being an unravel net. We then have to use a weaker criterion for merging nets. For this purposes we consider a special class of unravel nets. We take unravel nets that can be decomposed in smaller parts satisfying some properties, a sort of generalization of multi-clock nets (see the subsection 2.2). In these nets if a transition puts a token in a place s of a partition ν then it must have consumed a token from another place (at least one) of the same partition. This is formalized in the following definition:

Definition 5.6. *Let* $\mathsf{R} = (R, l)$ *be a labeled unravel net, where* $R = \langle S, T, F, \mathsf{m} \rangle$. *Let* $S_1, \ldots, S_k$ *be a partition of* S *such that* $\forall t \in T, \forall s \in t^\bullet$. $s \in S_i$ *implies that* $\exists s' \in S_i$ *and* $s' \in {}^\bullet t$, *for some* $i \in \{1, \ldots, k\}$, *and for each* $i \in \{1, \ldots, k\}$, $R_i = \langle S_i, {}^\bullet S_i \cup S_i{}^\bullet, F_i, \mathsf{m}_i, l \rangle = R|_{S_i}$ *is a connected Petri net. We call* R *a labeled* partitioned *unravel net and we denote the partition as* $\nu : S \to \{1, \ldots, k\}$ *such that* $S_i = \nu^{-1}(i)$.

We observe that $R = \langle \bigcup_i S_i, \bigcup_i ({}^\bullet S_i \cup S_i{}^\bullet), \bigcup_i F_i, \bigcup_i \mathsf{m}_i, \bigcup_i l_i \rangle$. The notion of partition is clearly inspired to the one of multi-clock net and the unique requirement we pose is that a token is produced in a place belonging to a transition, then there must be a place belonging to the same partition in the preset, which implies that a partition cannot *move* just receiving a token from another one, but is should be able to move independently.

Definition 5.7. *Let* $(\mathsf{R}, \nu) = ((R, l), \nu)$, *with* $R = \langle S, T, F, \mathsf{m} \rangle$, *be a labeled partitioned unravel net. We say that* $\delta : S \to \mathbb{N}$ *is a* locally strictly increasing measure

if for each state $X \in \mathsf{St}(R)$*, and each pair of places* s, s' *of the net* $R|_{[\![X]\!]}$*, it holds that* $s \leqslant s' \wedge \nu(s) = \nu(s') \Rightarrow \delta(s) < \delta(s')$.

Propositions 5.7, 5.8 and 5.9 can be lifted to partitioned unravel nets and locally strictly increasing measures.

Proposition 5.10. *Let* $(\mathsf{R}, \nu) = ((R, l), \nu)$ *with* $R = \langle S, T, F, \mathsf{m}\rangle$ *be a labeled partitioned unravel net, and let* $\delta : S \to \mathbb{N}$ *be a locally strictly increasing measure, then* $\forall s, s' \in S$ *such that* $(s^\bullet \cap {}^\bullet s' \neq \varnothing) \wedge (\nu(s) = \nu(s'))$ *it holds* $\delta(s) < \delta(s')$.

Proof. Let $t \in s^\bullet \cap {}^\bullet s'$. In any subnet $R|_X$ that contains t (such X exists because each transition is executable in an unravel net) $s < s'$, then, by the hypothesis $\nu(s) = \nu(s')$ and by Def. 5.7, $\delta(s) < \delta(s')$. ■

Proposition 5.11. *Let* $(\mathsf{R}, \nu) = ((R, l), \nu)$ *with* $R = \langle S, T, F, \mathsf{m}\rangle$ *be a labeled partitioned unravel net, and let* $s_1 s_2 \cdots s_n\}$ *a sequence of places such that* $\forall i < n, (s_i^\bullet \cap {}^\bullet s_{i+1} \neq \varnothing) \wedge \nu(s_i) = \nu(s_{i+1})$, i.e. *there is a path from* s_1 *to* s_n*. Let* $\delta : S \to \mathbb{N}$ *be a locally strictly increasing measure, then* $\delta(s_1) < \delta(s_n)$.

Proposition 5.12. *Let* $(\mathsf{R}, \nu) = ((R, l), \nu)$ *with* $R = \langle S, T, F, \mathsf{m}\rangle$ *be a labeled partitioned unravel net, and let* $\delta : S \to \mathbb{N}$ *be a locally strictly increasing measure, then for each partition* S_i *of* S *it holds* $R|_{S_i}$ *is acyclic.*

Proof. Assume there is a cycle $s_1 s_2 \cdots s_n s_1$, from Prop.5.11 it would happen that $\delta(s_1) < \delta(s_1)$, which is an absurd. ■

The following theorem shows that if we have a partitioned unravel net, where we defined a locally strictly increasing measure, we can merge it and obtain faithfully another unravel net.

Theorem 5.2. *Let* $(\mathsf{R}, \nu) = ((R, l), \nu)$ *with* $R = \langle S, T, F, \mathsf{m}\rangle$ *be a labeled partitioned unravel net, let* $\delta : S \to \mathbb{N}$ *be a* locally strictly increasing measure *and let* $\sim$ *be the equivalence relation induced by* δ, i.e. $s \sim s'$ *iff* $(\delta(s) = \delta(s') \wedge l(s) = l(s') \wedge \nu(s) = \nu(s'))$ *or* $s = s'$*. Then the resulting compact labeled net* $\overline{\mathsf{R}} = (\overline{R}, \overline{l})$ *is a labeled unravel net.*

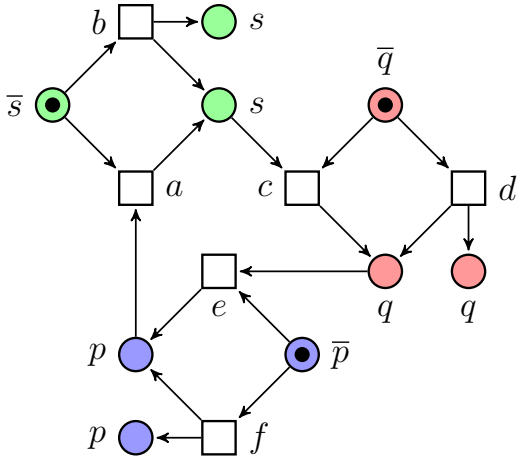


Figure 5.4: A partitioned unravel net.

Proof. Suppose there exists a $\mathsf{fs}\ \sigma = m\,[t_0\rangle\, m_1\,[t_1\rangle \cdots [t_{i-1}\rangle\, m_i\,[t_i\rangle \cdots [t_{j-1}\rangle\, m_j\,[t_j\rangle \cdots \in \mathcal{R}_m^{\overline{R}}$ such that $m_i \cap m_j \neq \varnothing$ and $\exists [s]_\sim \in m_i \cap m_j$ such that $m_k([s]_\sim) = 0$ for $i < k < j$, *i.e.* $[s]_\sim$ is marked twice in σ.

Since (R, ν) is a partitioned then, $\forall s' \in [s]_\sim$ we can find in ${}^\bullet t_{j-1}$ an equivalence class with a representant s'' such that $\nu(s'') = \nu(s')$, hence $\delta(s'') < \delta(s')$. By repeating the process from s'' till $\overline{s} \in t_i{}^\bullet$ we eventually find again $s' \in {}^\bullet t_i$ with $\nu(\overline{s} = \nu s'$. The chain of places $\overline{S} = s's'' \cdots \overline{s}s'$ satisfies the hypotheses of Prop. 5.11, then $\delta(s') < \delta(s')$ which is impossible, then there is no firing sequence in (R, ν) which allows to mark twice a place. This prove that (R, ν) restricted to a state is safe and acyclic. That subnet is also causal because a place with two incoming arcs would violate the safeness or the acyclicity, then it is as unravel net. ■

When the measure is not strictly increasing or locally strictly increasing we may still obtain an unravel net, but sometimes at the price of *enriching* it in order to forbid certain unwanted executions in the compact version. The notion of measure we consider now is inspired from the one of occurrence depth.

Definition 5.8. *Given a labeled unravel net* $\mathsf{R} = (R, l)$ *with* $R = \langle S, T, F, \mathsf{m}\rangle$*, we say that a measure* $\delta : S \to \mathbb{N}$ *is* homogeneous *iff for each* $X \in \mathsf{St}(R)$ *and each subset of places* $\hat{S}$ *of* $R|_X = \langle S', X, F', \mathsf{m}\rangle$ *such that there exists a label* a *such that* $l^{-1}(\mathsf{a}) \cap S' = \hat{S}$*, it holds that* $\hat{S}'$ *can be totally ordered with respect to the reflexive*

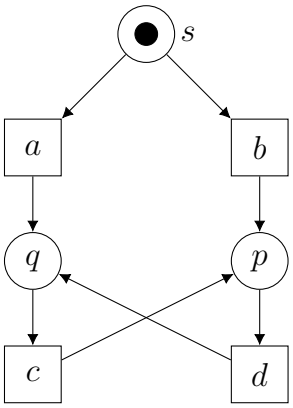


Figure 5.5: A labeled net.

and transitive closure of F', $\delta(\hat{S}') = \{1, \ldots, |\hat{S}'|\}$ *and* $s \leqslant s' \;\Rightarrow\; \delta(s) \leqslant \delta(s')$.

An homogeneous measure on causal nets is the token count of merged process. Again, as done before, we may introduce an equivalence relation which is based on an homogeneous measure δ by stipulating that $s \sim s'$ iff either $s = s'$ or $(l(s) = l(s') \;\wedge\; \delta(s) = \delta(s'))$. As the measure is an homogeneous one, we do not have to require that the two places are incompatible as it is implied by the definition of the measure itself. When compacting using this merging relation the result may be not an unravel net (a merged process with executable cycles, for instance). However this net may be turned into an unravel one without losing behaviors of the original net by adding some places which sole purpose is to forbid unwanted executions.

Take an unravel net $\mathsf{R} = (\langle S, T, F, \mathsf{m}\rangle, l)$ and an homogeneous measure δ on S. We can add to the net $R = \langle S, T, F, \mathsf{m}\rangle$ a set of places $S_{ng} = \{(l(s), \delta(s), ng) \mid s \in S\} \cup \{(l(s), 0, ng) \mid s \in S \backslash \mathsf{m}\}$, and connect them to the transitions in T as follows: $F_{ng}((l(s), n, ng), t) = 1$ whenever $\exists s' \in t^\bullet.\ l(s) = l(s')$ and $\delta(s') = n + 1$, and $F_{ng}(t, (l(s), n, ng)) = 1$ whenever $\exists s' \in t^\bullet.\ l(s) = l(s')$ and $\delta(s') = n$; finally the places $(l(s), 0, ng)$ are initially marked as well as $(l(s), 1, ng)$ if $b \in \mathsf{m}$ (and are the multiset m_{ng}).

We call these places *no-gap* as in the case that the δ is precisely the token count and $\mathsf{R} = (R, l)$ is a branching process of a safe net N, they assure that the tokens in

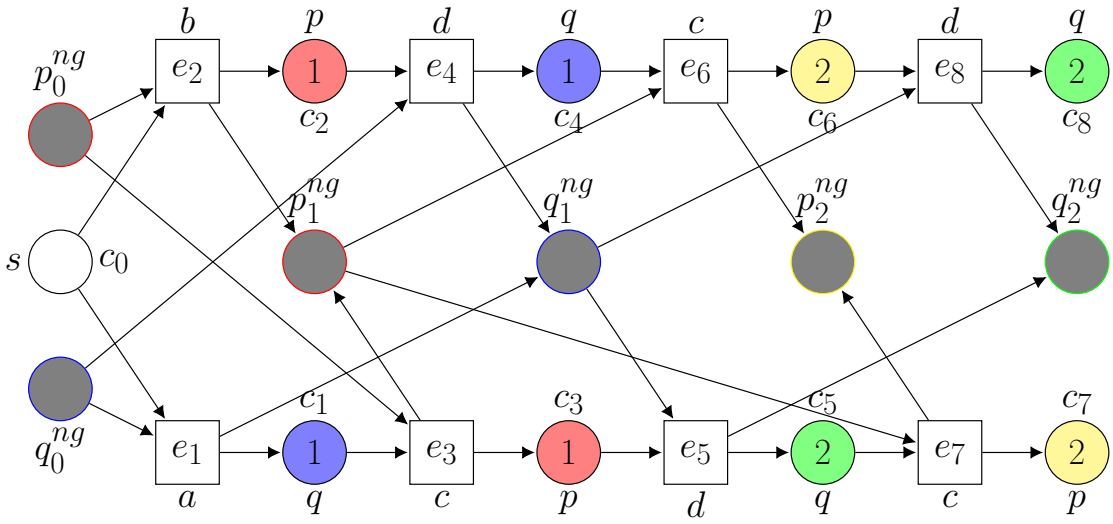


Figure 5.6: The enriched branching process of the net in Fig. 5.5.

a place of the original net N are *produced* in the proper sequence. The net obtained by adding these new places S_{ng}, namely $\mathsf{Ng}(R) = \langle S \cup S_{ng}, T, F \cup F_{ng}, \mathsf{m} + \mathsf{m}_{ng} \rangle$, is an unravel net such that to each $\mathsf{fs}\ \sigma$ of $\mathsf{Ng}(R)$ a $\mathsf{fs}\ \sigma'$ of R corresponds and they are such that $run(\sigma) = run(\sigma')$ but also the *vice versa* holds, thus to each $\mathsf{fs}\ \hat{\sigma} \in \mathcal{R}^R_{\mathsf{m}}$ a $\mathsf{fs}\ \hat{\sigma}' \in \mathcal{R}^{\mathsf{Ng}(R)}_{\mathsf{m}+\mathsf{m}_{ng}}$ corresponds such that $run(\hat{\sigma}) = run(run(\sigma)')$, hence both unravel nets have exactly the same states, which means that this enriching does not change the behaviors of the net. The following theorem states that merging an enriched unravel net generates an unravel net. This somehow generalizes Theorem 4.1.

Theorem 5.3. *Let* $\mathsf{R} = (R, l)$ *be a labeled unravel net, where* $R = \langle S, T, F, \mathsf{m} \rangle$. *Let* $\mathsf{Ng}(\mathsf{R}) = (\mathsf{Ng}(R), \mathsf{Ng}(l))$ *where* $\mathsf{Ng}(l)(s) = l(s)$ *if* $s \in S$ *and* $l((s, n, ng)) = l(n)$ *obtained with respect to an homogeneous measure* δ. *Let* $\sim$ *be the equivalence relation induced by this measure on the places in* S. *Then* $\overline{\mathsf{Ng}(\mathsf{R})} = (\overline{\mathsf{Ng}(R)}, \overline{\mathsf{Ng}(l)})$ *is an unravel net.*

Proof. Merging relations preserve safeness. The proof that $\overline{\mathsf{Ng}(\mathsf{R})}$ is indeed an unravel net is the same as the one of Theorem 4.1. ■

As a corollary of this theorem we have the following.

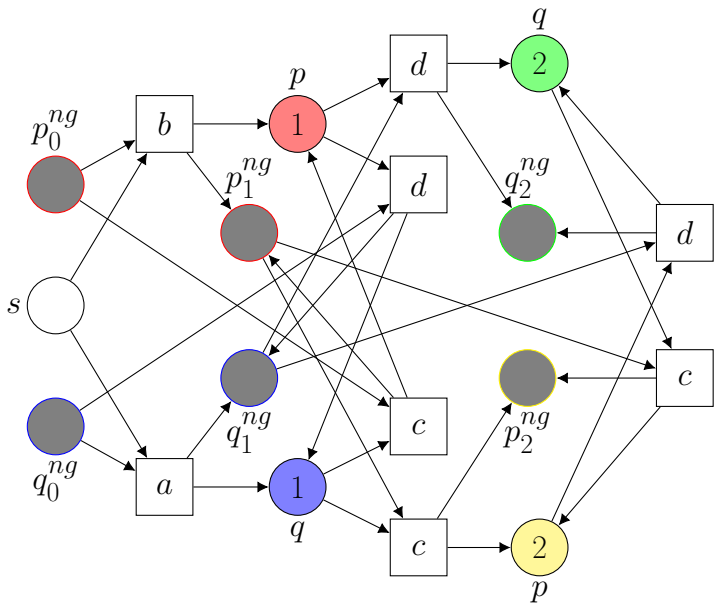


Figure 5.7: The resulting unravel net of the compaction of $\mathsf{Ng}(C)$ in Fig. 2.1

Corollary 5.1. *Let* $\mathsf{C} = (C, p)$ *be a branching process of the safe net* N*, where* $C = \langle B, E, F, \mathsf{m}\rangle$*. Let* $\mathsf{Ng}(\mathsf{C}) = (\mathsf{Ng}(C), \mathsf{Ng}(p))$ *be the unravel net obtained by applying* Ng *to* C *and* p *and consider the equivalence relation* $\sim$ *induced by* $\sim_{tok}$*, where* $\sim_{tok}$ *is defined on conditions in* B*.* $\overline{\mathsf{Ng}(\mathsf{C})} = (\overline{\mathsf{Ng}(C)}, \overline{\mathsf{Ng}(p)})$ *in a labeled unravel net and furthermore* $\mathsf{Ng}(\mathsf{C})|_{B_{\sim_{tok}}}$ *is a merging process of* N*, where* $B_{\sim_{tok}}$ *are the merged resource conditions.*

Chapter 6

Contextual unravel nets and event structures

Event structures are one of the main ways to represent computations in service oriented computations, thus the quest for event structures where the new kind of dependencies may be easily represented.

Consider a *seller* behaving as follows. When he receives a request for delivering a good, he sends a confirmation if the good is immediately available, but if it is not (hence the `not-available` event happens) he sends a confirmation only after being sure that the good is available in some other places and he may eventually get it. Furthermore he delivers the good only after receiving the payment unless the one requesting the good is a premium user. In this scenario the event `confirm` may happens after the event `request` has happened unless the `not-available` event has happened. In this case the event `confirm` *depends* not only on `request`, but also on the event `available` signalling that the good is somewhere available. The dependency pattern of `confirm` may *augment* in presence of a certain event (in this case `not-available`). The event `deliver` depends on the event `pay`, but if the event `premUser` is present then this dependency is dropped.

Event structures have generally a relation to represent the *dependencies* among events and another relation to stipulate which events cannot happen in the same

computation under certain conditions. A computation is then a subset of events such that there is no pair of events in conflict and each event is such that all of its causes are contained in the configuration. The dependencies are often represented as a partial order or a relation that should be a partial order when confined to configuration, like in flow event structures.

In this chapter we give a Petri net counterpart to the notions of shrinking and growing causality that, as the example shows, can be one of the relevant ingredients for service oriented computations. The authors in [AKPN15a] state that in the usual notion of event structures the dependencies have a *static* flavour, whereas, in the case of shrinking and growing causality, the dependencies to be dropped or added give a more *dynamic* account. However in the shrinking and growing causality relations it is somehow hardwired which dependencies are involved. Thus an event may have several *histories*, depending on the combination of modifiers that have happened so far. Our idea is rather simple. We introduce a transition (with the same label) for each possible history (similarly to what happens when considering labelled causal nets like the notion of *unfolding* [Win87] or *branching processes* [Eng91]) and we use contextual arcs to understand which is the set of modifiers that have happened so far. The kind of nets we devise should still have some characteristics similar to the ones of causal nets, hence we require that the restriction of the net to a specific subset of transitions (a computation) gives an acyclic net, *i.e.* a net where dependencies are easily understandable and conflicts are absent. We call this notion *unravel net*.

The notion of unravel net with contextual arcs turns out to be powerful enough to represent shrinking and dynamic causality, but it should be stressed that we actually give an implementation of how the executions of the event structures may be performed. For this reason we focus on traces rather than on configurations.

By exploiting the relationship between dynamic event structures and labeled unravel nets with contextual arcs where the contextual arcs are not inhibitor ones,

we remain in the classes of nets where many properties can be easily and efficiently verified.

6.1 Contextual Petri nets

We recall the main definitions about labeled Petri nets with contextual arcs.

Definition 6.1. *A* Petri net *with contextual arcs is a 5-tuple* $N = \langle S, T, F, C, \mathsf{m}\rangle$, *where* S *is a set of* places *and* T *is a set of* transitions *(with* $S \cap T = \varnothing$*),* $F \subseteq (S \times T) \cup (T \times S)$ *is the* flow *relation,* $C \subseteq (T \times S)$ *is the* context *relation and* $\mathsf{m} \in \mu S$ *is called the* initial marking.

We require that $(t, s) \in C$ implies $(t, s) \notin F$. Contextual Petri nets are depicted as usual: places are circles, transitions are boxes, the flow relation is represented by arcs from x to y whenever $(x, y) \in F$ and the read arcs are depicted as a straight lines.

Since we are not interested anymore on identifying places, the labeling mapping will be defined for transitions only.

Definition 6.2. *A* labeled *Petri net over* L *is the pair* $\mathsf{N} = (N, l)$ *where* $N = \langle S, T, F, C, \mathsf{m}\rangle$ *is a Petri net with contextual arcs and* $l\colon T \to \mathsf{L}$ *is a total labeling mapping.*

Given a transition t, its *context* is $\underline{t} = \{s \in S \mid (t, s) \in C\}$. Also $\underline{t}$ may be seen as a multiset on S: $\underline{t}(s) = 1$ whenever $(t, s) \in C$ and $\underline{t}(s) = 0$ otherwise. A transition $t \in T$ is enabled at a marking $m \in \mu S$, denoted with $m\,[t\rangle$, whenever ${}^\bullet t \subseteq m$ and $\underline{t} \subseteq m$. A transition t enabled at a marking m can *fire* and its firing produces the marking $m' = m - {}^\bullet t + t^\bullet$. The firing of a transitions t at a marking m is denoted with $m\,[t\rangle\, m'$. We stress that the context is used only to check whether or not a transition is enabled at a given marking. ${}^\bullet t + \underline{t} \neq \varnothing$ (which means that no transition may fire *spontaneously*).

The other notions like firing sequences and reachable markings are the obvious generalizations of the ones on safe nets without contextual arcs.

6.2 Contextual Unravel Nets

Definition 6.3. *A* contextual unravel net *(c-unravel net)* $N = \langle S, T, F, C, \mathsf{m}, l \rangle$ *over a set of label* L *is a safe occurrence net such that*

1. *for each state* $X \in \mathsf{St}(N)$ *the net* $N|_X^{S'}$ *is acyclic and conflict-free, where* $S' = {}^\bullet X \cup X^\bullet \cup \underline{X}$,

2. *for all* $t_1, t_2 \in T$, *if* $t_1 \# t_2$ *and* $l(t_1) \neq l(t_2)$ *then for each* $t \in l^{-1}(l(t_1))$ *and for each* $t' \in l^{-1}(l(t_2))$ *it holds that* $t \# t'$,

3. *for each* $\mathsf{e} \in l(T)$ *there exists a unique place* s *such that* $\{s\} \subseteq \bigcap_{t \in l^{-1}(\mathsf{e})} {}^\bullet t$ ${}^\bullet s = \varnothing$ *and* $\mathsf{m}(s) = 1$, *and we denote that place with* ${}^\diamond\mathsf{e}$, *and*

4. *for each* $\mathsf{e} \in l(T)$ *there exists a unique place* s *such that* $\{s\} \subseteq \bigcap_{t \in l^{-1}(\mathsf{e})} t^\bullet$, $s^\bullet = \varnothing$, $\mathsf{m}(s) = 0$ *and* ${}^\bullet s = l^{-1}(\mathsf{e})$, *and we denote that place with* $\mathsf{e}^\diamond$.

The whole unravel net is not constrained to be either acyclic or conflict-free, but each of its execution gives an acyclic and conflict-free net. Condition (2) guarantees that if two transitions are in conflict and have different labels, then any pair of transitions with the same labels are in conflict as well. The other conditions guarantee that two equally labeled transitions are in conflict, each transition can happen just once and equally labeled transitions have some common places in their preset. The two final conditions ensure that all equally labelled transitions are in conflict (as they share a place in the preset which is initially marked and that cannot be filled again), and all of them put in a place (where no other transition beside the ones bearing that specific label can put a token). This place may be redundant, unless it is connected to a transition with a contextual arc.

As equally labeled transitions are in conflict in an unravel net all the transitions in a firing sequence have different labels. The sequence of labels of a fs form a *trace* of the net. Formally,

Definition 6.4. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be an unravel net over a set of labels L. Let $\sigma \in \mathcal{R}^N_{\mathsf{m}}$ with $\sigma = \mathsf{m}\,[t_1\rangle m_1\,[t_2\rangle m_2 \cdots m_{n-1}\,[t_n\rangle m_n$. Then a* trace *of N is the sequence of labels $l(t_1 t_2 \cdots t_n)$ and it is denoted by $run(\sigma)$. The set of traces of a net is $\mathsf{Tr}(N) = \{w \in \mathsf{L}^* \mid \exists \sigma \in \mathcal{R}^N_{\mathsf{m}}.\ run(\sigma) = w\}$*

Configurations are obtained by forgetting the sequence, as each state is obtained by a firing sequence. Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be an unravel net over a set of labels L. Let $X \in \mathsf{St}(N)$, then $\mathsf{X} = \{l(t) \mid t \in X\}$ is a *configuration* of N. The set of configuration of a net is denoted with $\mathsf{Conf}(N)$.

6.3 Beyond prime and bundle event structures

Event structures usually model *concurrency systems* by defining relationships among events such as *causality* and *conflict*. We recall the definitions of various kind of event structures (not of all possible variants). In order to describe the state of those systems we will adopt the one of events trace which, besides some peculiar cases, is enough to retrieve the the usual notion of configuration. Consider a set of events E and let $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ be a sequence of distinct events in E and with overloading of notation with ϵ we denote the *empty* sequence. With $\overline{\rho}$ we denote the set $\{\mathsf{e}_1, \ldots, \mathsf{e}_n\}$ and the set associated to the empty sequence is the empty set. Given a sequence ρ of events, its length is $|\overline{\rho}|$ and it is denoted with $len(\rho)$ for each $1 \leqslant i \leqslant len(\rho)$ with ρ_i we denote the sequence $\mathsf{e}_1 \cdots \mathsf{e}_i$, and with ρ_0 we denote the empty sequence.

In Def. 2.12, (1) is the *axiom of finite causes* which states that events cannot depend on an infinite number of other events, and (2), contains the *conflict heredity* property, which says if two events are in conflict then each one conflicts with all of

the other one's successors.

In the following we will show several types of event structures defined as extensions of PES. Since some of them can express a sort of dynamism in the structure (e.g. adding or dropping causal dependencies), clauses (1) and (2) will be hard to maintain. To cope with that limitation we will consider a different definition of PES which is equivalent to Def. 2.12 if finite configurations are considered.

6.3.1 Simple prime event structures

We start by defining the notion of *simple prime event structure* we will use in this paper following [AKPN15a]. We consider only finite event structures, thus the set of events should be always considered finite.

Definition 6.5. *A* simple prime event structure (SPES) *is a triple* $P = (\mathsf{E}, \rightarrow, \#)$*, where* E *is a finite set of* events *and* $\rightarrow$*,* $\#$ *are two not intersecting binary relations on* E *called* causality relation *and* conflict relation *respectively, such that* $\#$ *is irreflexive and symmetric.*

This notion differ substantially from the one given in [Win87], but both have the same configurations and traces, as we are considering finite event structures, as we will show.

Given a SPES $P = (\mathsf{E}, \rightarrow, \#)$ and an event $\mathsf{e} \in \mathsf{E}$ with $\mathrm{ic}(\mathsf{e})$ we denote the set $\{\mathsf{e}' \mid \mathsf{e}' \rightarrow \mathsf{e}\}$. A sequence of events $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ is a trace whenever $\overline{\rho}$ is *conflict free, i.e.* for each $\mathsf{e}_i, \mathsf{e}_j \in \overline{\rho}$, $\mathsf{e}_i \neq \mathsf{e}_j \;\Rightarrow\; \neg(\mathsf{e}_i \# \mathsf{e}_j)$ and for each $i \leqslant n$, $\mathrm{ic}(\mathsf{e}_i) \subseteq \overline{\rho_{i-1}}$. Given a trace ρ the associated configuration is $\overline{\rho}$.

6.3.2 Dual event structures

Definition 6.6. *A* Dual Event Structure (DES) *is a triple* $\delta = (E, \#, \mapsto)$*, where* E *is a set of events,* $\# \subseteq E^2$ *is an irreflexive symmetric relation (the* conflict *relation), and* $\mapsto \subseteq \mathcal{P}(E) \times E$ *is the* enabling *relation.*

The definitions of traces and configurations are the same as the ones of BES(Def. 3.4 and 3.5).

We now recall some notions of event structure where causality may change.

6.3.3 Shrinking Event Structures

Given a PES it is possible to add a suitable relation to model the removal of causal dependencies [AKPN15a]. The idea is to introduce a ternary relation stipulating that the happening of a specific event (the *modifier*) allows to drop a specific cause for another event (the *target*).

Definition 6.7. *A* Shrinking Causality Event Structure (SES) *is the quadruple* $\gamma = (\mathsf{E}, \rightarrow, \#, \triangleright)$ *where* $(\mathsf{E}, \rightarrow, \#)$ *is a* PES *and* $\triangleright \subseteq \mathsf{E} \times \mathsf{E} \times \mathsf{E}$ *is the* shrinking causality *relation such that* $[\mathsf{e} \rightarrow \mathsf{e}''] \triangleright \mathsf{e}'$ *implies* $\mathsf{e} \rightarrow \mathsf{e}''$ *for all* $\mathsf{e}, \mathsf{e}', \mathsf{e}'' \in \mathsf{E}$.

The only extra requirement is that to have a cause to drop for an event, this cause should be hardwired in the causality relation $\rightarrow$. For $[\mathsf{c} \rightarrow \mathsf{t}] \triangleright \mathsf{m}$ we call m the modifier, t the target and c the *contribution*, and we denote with $[\mathsf{c} \rightarrow \mathsf{t}]\triangleright$ the set of all modifiers dropping $\mathsf{c} \rightarrow \mathsf{t}$. The set of dropped causes of an event can be defined w.r.to a specific history, *i.e.* a set of events, by the function $\mathrm{dc}\colon \mathbf{2}^{\mathsf{E}} \times \mathsf{E} \rightarrow \mathbf{2}^{\mathsf{E}}$, which is defined as follows: $\mathrm{dc}(H, \mathsf{e}) = \{\mathsf{e}' \mid \exists \mathsf{d} \in H.\ [\mathsf{e}' \rightarrow \mathsf{e}] \triangleright \mathsf{d}\}$. As for PES, we have a function $\mathrm{ic}\colon \mathsf{E} \rightarrow \mathbf{2}^{\mathsf{E}}$ defined in the same way, *i.e* $\mathrm{ic}(\mathsf{e}) = \{\mathsf{e}' \mid \mathsf{e}' \rightarrow \mathsf{e}\}$, and we say that $\mathrm{ic}(\mathsf{e})$ are the *initial* causes of the event e.

Definition 6.8. *Traces for a* SES *are defined as follows. Let* $\gamma = (\mathsf{E}, \rightarrow, \#, \triangleright)$ *be a* SES. *A* trace *of* γ *is a sequence of distinct events* $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ *with* $\overline{\rho} \subseteq \mathsf{E}$ *such that:*

1. $\overline{\rho}$ *is conflict-free and*

2. $\forall 1 \leqslant i \leqslant \mathit{len}(\rho).\ (\mathrm{ic}(\mathsf{e}_i) \setminus \mathrm{dc}(\overline{\rho_{i-1}}, \mathsf{e}_i)) \subseteq \overline{\rho_{i-1}}$.

The relevant difference with the notion of trace for PES is just that in the enabling of each event, depending on the modifiers happened so far (hence the role of the history), the set of the immediate causes may shrink.

Example 6.1. *Consider the following* SES*, with just three events* a, b *and* c*, with* $\mathsf{b} \to \mathsf{c}$ *and* $[\mathsf{b} \to \mathsf{c}] \rhd \mathsf{a}$*.* a *is the modifier for the target* c *and its happening has the effect that the cause* b *may be dropped. The maximal traces are:* acb*,* abc*,* bca *and* bac*. It is worth to notice that in all but the trace* bca *the event* c *has no predecessor.*

6.3.4 Growing Event Structures

Opposite to shrinking causality there is the notion of growing causality [AKPN15a], in which a dependency between two events is added after the execution of a third event.

Definition 6.9. *A* growing causality event structure (GES) *is the quadruple* $\delta = (\mathsf{E}, \to, \#, \blacktriangleright)$ *where* $(\mathsf{E}, \to, \#)$ *is a* PES *and* $\blacktriangleright \subseteq \mathsf{E} \times \mathsf{E} \times \mathsf{E}$ *is the* growing causality *relation such that* $\forall \mathsf{e}, \mathsf{e}', \mathsf{e}'' \in \mathsf{E}.\ \mathsf{e}' \blacktriangleright [\mathsf{e} \to \mathsf{e}''] \implies \neg(\mathsf{e} \to \mathsf{e}'')$*.*

Here the requirement is that a dependency can be added only if it is not hard-wired in $\to$. For $\mathsf{m} \blacktriangleright [\mathsf{c} \to \mathsf{t}]$ we call m the *modifier*, t the *target* and c the *contribution*, and we denote with $\blacktriangleright [\mathsf{c} \to \mathsf{t}]$ the set of all modifiers adding $\mathsf{c} \to \mathsf{t}$. The set of added causes of an event has to be defined *w.r.t* a specific history, *i.e.* a set of events, and it is using the function $\mathrm{ac}\colon \mathbf{2}^{\mathsf{E}} \times \mathsf{E} \to \mathbf{2}^{\mathsf{E}}$, which is defined as follows: $\mathrm{ac}(H, \mathsf{e}) = \{\mathsf{e}' \mid \exists \mathsf{d} \in H.\ \mathsf{d} \blacktriangleright [\mathsf{e}' \to \mathsf{e}]\}$. The initial causes of an event are defined as usual.

Definition 6.10. *Let* $\delta = (\mathsf{E}, \to, \#, \blacktriangleright)$ *be a* GES*. A* trace *of* δ *is a sequence of distinct events* $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ *with* $\overline{\rho} \subseteq \mathsf{E}$ *such that*

1. $\overline{\rho}$ *is conflict-free and*
2. $\forall 1 \leqslant i \leqslant \mathit{len}(\rho).\ (\mathrm{ic}(\mathsf{e}_i) \cup \mathrm{ac}(\overline{\rho_{i-1}}, \mathsf{e}_i)) \subseteq \overline{\rho_{i-1}}$.

Example 6.2. *Consider just three events* a, b *and* c*, and the unique non empty relation is* $\mathsf{a} \blacktriangleright [\mathsf{b} \to \mathsf{c}]$. a *is the modifier for the target* c *and its happening has the effect that the cause* b *should be added. The maximal traces are:* abc*,* bac*,* cab *and* cba*. It is worth to notice that in all but the traces* abc*,* bac *the event* c *must happen after* b*.*

6.3.5 Dynamic causality event structure

In [AKPN15a] it is shown that neither the shrinking causality relation can be expressed with the growing causality relation or not the converse, hence SES and GES are incomparable. Thus the two notions of shrinking and growing causality can be put together in a definition.

Definition 6.11. *A* dynamic causality event structure (DCES) *is a quintuple* $\Sigma = (\mathsf{E}, \to, \#, \rhd, \blacktriangleright)$*, where* $(\mathsf{E}, \to, \#)$ *is a* PES $\rhd \subseteq \mathsf{E} \times \mathsf{E} \times \mathsf{E}$ *is the* shrinking causality *relation, and* $\blacktriangleright \subseteq \mathsf{E} \times \mathsf{E} \times \mathsf{E}$ *is the* growing causality *relation, and are such that for all* $\mathsf{e}, \mathsf{e}', \mathsf{e}'' \in \mathsf{E}$

1. $[\mathsf{e} \to \mathsf{e}''] \rhd \mathsf{e}' \wedge \nexists \mathsf{m} \in \mathsf{E}.\ \mathsf{m} \blacktriangleright [\mathsf{e} \to \mathsf{e}''] \Longrightarrow \mathsf{e} \to \mathsf{e}''$*,*

2. $\mathsf{e}' \blacktriangleright [\mathsf{e} \to \mathsf{e}''] \wedge \nexists \mathsf{m} \in \mathsf{E}.[\mathsf{e} \to \mathsf{e}''] \rhd \mathsf{m} \Longrightarrow \neg(\mathsf{e} \to \mathsf{e}'')$*, and*

3. $\forall \mathsf{e}, \mathsf{e}' \in \mathsf{E}.\ \nexists \mathsf{m}, \mathsf{a} \in \mathsf{E}.\ \mathsf{a} \blacktriangleright [\mathsf{e} \to \mathsf{e}'] \rhd \mathsf{m}$.

For detailed comments on this definition we refer to [AKPN15a], it should be observed, however, that the definition we consider here is slightly less general of the one presented there, as we add a further condition, the last one, which is defined in the [AKPN15b] and it does not allow that the same contribution can be added and removed by two different modifiers. These are called in [AKPN15b] *single state dynamic causality event structures* and rule out the fact that some causality (or absence of) depends on the order of modifiers. Conditions (1) and (2) simply rephrase the conditions under which the shrinking and growing relations

are defined: in the case of the shrinking relation the dependency should be present and in the case of the growing not; condition (3) says that if a dependency is added then it cannot be removed.

The traces are defined accordingly.

Definition 6.12. *Let* $\Sigma = (\mathsf{E}, \rightarrow, \#, \triangleright, \blacktriangleright)$ *be a* DCES. *A* trace *of* Σ *is a sequence of distinct events* $\rho = \mathsf{e}_1 \cdots \mathsf{e}_n$ *with* $\overline{\rho} \subseteq \mathsf{E}$ *such that*

1. $\overline{\rho}$ *is conflict-free and*

2. $\forall 1 \leqslant i \leqslant \mathit{len}(\rho).\ ((\mathrm{ic}(\mathsf{e}_i) \cup \mathrm{ac}(\overline{\rho_{i-1}}, \mathsf{e}_i)) \setminus \mathrm{dc}(\overline{\rho_{i-1}}, \mathsf{e}_i)) \subseteq \overline{\rho_{i-1}}$.

Example 6.3. *Consider the set of events* $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{e}\}$, *with* $\mathsf{b} \rightarrow \mathsf{c}$, $[\mathsf{b} \rightarrow \mathsf{c}] \triangleright \mathsf{a}$ *and* $\mathsf{d} \blacktriangleright [\mathsf{e} \rightarrow \mathsf{c}]$. a *and* d *are the modifiers for the target* c, *the happening of* a *has the effect that the cause* b *may be dropped, and the one of* d *that the cause* e *should be added for* c. *If the prefix of the trace is* bc *(the target* c *is executed before of one of its modifiers* a *and* d*) then the final part of the trace is any combination of* a, e *and* d. *If the modifier* a *is executed before* c *then we have the traces* ac *followed by any combination of the remaining three events,* abcde *and* abced *and finally if the two modifiers happen before* c *we have* adecb *or* adebc *and similarly we have* daecb *and* daebc. *Clearly, after the happening of the two modifiers in any order, it not prescribed that* c *happens immediately, as it may happen after* b *which is not any longer a cause for it.*

We now recast in this setting the example discussed in the introduction.

Example 6.4. *The events of the seller example illustrated in the introduction are* c *corresponding to* `confirm`, r *to* `request`, n *to* `not-available`, a *to* `available` *which means available somewhere else,* p *to* `pay`, d *to* `deliver`, i *that means that the good is present in stock and finally* u *to* `premUser`. *The event structure has just a pair of events in conflict, namely* i *and* n *and the various relations are: the causal relation is* $\mathsf{r} \rightarrow \mathsf{i}, \mathsf{r} \rightarrow \mathsf{a}, \mathsf{i} \rightarrow \mathsf{c}, \mathsf{p} \rightarrow \mathsf{d}, \mathsf{c} \rightarrow \mathsf{d}$ *and* $\mathsf{u} \rightarrow \mathsf{d}$, *the shrinking*

relation is $[\mathsf{p} \to \mathsf{d}] \rhd \mathsf{u}$, $[\mathsf{i} \to \mathsf{c}] \rhd \mathsf{n}$ *and* $[\mathsf{u} \to \mathsf{d}] \rhd \mathsf{p}$ *and finally the growing relation is* $\mathsf{n} \blacktriangleright [\mathsf{a} \to \mathsf{c}]$. *Once that the event* r *happens then there is choice: the good is present (*i*) or not (*n*). In the former situation the confirm message can be issued (*c*) and after that the good can be delivered, either after a payment* p *or because the one issuing the request is a premium user (*u*). A trace could be* ricpd *or* ricud. *A premium user can also pay, but this has no influence on the delivering of the good. In the latter situation (the good is not in stock) the* c *depends on the availability of the good at some place (*$\mathsf{n} \blacktriangleright [\mathsf{a} \to \mathsf{c}]$*) and it should not depend any longer from* i *(*$[\mathsf{i} \to \mathsf{c}] \rhd \mathsf{n}$*). A trace is* rnacpd.

6.4 Unravel nets and event structures

In this section we relate unravel nets and event structures. We characterize some unravel nets and we show how to associate an event structure to them, and then we will directly consider DCES and associate to a DCES a suitable unravel net.

6.4.1 From unravel nets to event structures

Given an unravel net $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ over a set of labels L, a transition $t' \in T$, with $\mathsf{Pred}(t')$ we denote the set $\{t \in T \mid t^\bullet \cap {}^\bullet t' \neq \varnothing\}$. Two different labels e and e' in $l(T)$ are said to be in conflict, denoted $\mathsf{e} \bigstar \mathsf{e}'$ iff $\exists t \in l^{-1}(\mathsf{e})$ and $\exists t' \in l^{-1}(\mathsf{e}')$ and $t \# t'$. Observe that due to condition (b) of Def. 3.1 this is well defined.

We start characterizing unravel nets representing event structures where no modifier is present.

Definition 6.13. *Let* $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ *be an unravel net over the set of labels* L. *We say that* N *is* simple *whenever* l *is injective,* C *is empty and for each* $s \in S$. $|{}^\bullet s| \leqslant 1$.

Observe that a simple unravel net is indeed a causal net. To a simple unravel net a PES can be easily associated. The intuition is the expected one: to each

place s in the preset of a transition t labeled with an event e we associate a causal dependency for e, namely with the unique event associate to the transition ${}^\bullet s$. The proof of the following proposition is obvious.

Proposition 6.1. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a simple unravel net, then $\mathcal{E}_{si}(N) = (l(T), \rightarrow, \#)$ is a* SPES*, where the conflict relation is defined as follows: $\mathsf{e} \# \mathsf{e}'$ iff $\mathsf{e} \bigstar \mathsf{e}'$, and the causal relation is defined as follows: for each $\mathsf{e} \in l(T)$, for each $s \in {}^\bullet l^{-1}(\mathsf{e})$ we have $l({}^\bullet s) \rightarrow \mathsf{e}$.*

The following theorem states that a simple net and the associate SPES have the same traces.

Theorem 6.1. *Let $N = \langle S, T, F, \varnothing, \mathsf{m}, l\rangle$ be a simple unravel net and let $\mathcal{E}_{si}(N)$ be the associated* SPES*. Then w is a trace of N iff w is a trace of $\mathcal{E}_{si}(N)$.*

Proof. Observe that a simple unravel net is a special case of unravel net as it is a causal net. Hence the proof is as in Th. 3.1. ■

The requirements for an unravel net to be simple are quite strong. We start investigating what happens if we drop injectivity of the labeling mapping. Intuitively this means that we have various *implementations* of the same activity, as this one may happen due to various causes which are not any longer in the exclusive or relation.

Definition 6.14. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be an unravel net over the set of labels L. We say that N is* semi simple *whenever C is empty and l is not injective.*

Consider the net in Fig. 6.1, the transitions t_3, t_4 and t_5 represent the same activity c. This activity may be executed after a only (t_3), or after b only (t_5) and finally after both (t_4). Following the spirit of DES we do not implement a maximal causal semantics, where the instance of the transition with more *or*-causes is executed.

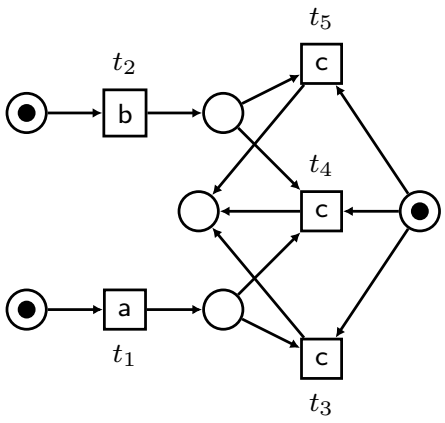


Figure 6.1: A semi simple unravel net

To this kind of net a DES is associated, as the following proposition shows. Observe that the bundle are formed in a slightly different way, with respect to the previous notion.

Proposition 6.2. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a semi simple unravel net, then $\mathcal{E}_{ss}(N) = (l(T), \#, \mapsto)$ is a* DES*, where*

- *the conflict relation is defined as follows: $\mathsf{e} \# \mathsf{e}'$ iff $\mathsf{e} \bigstar \mathsf{e}'$, and*
- *the bundle relation is defined as follows: for each $\mathsf{e} \in l(T)$, consider $T_{\mathsf{e}} = l^{-1}(\mathsf{e})$, then*
  - *for each $s \in \bigcap_{t \in T_{\mathsf{e}}} {}^\bullet t$ we have $l({}^\bullet s) \mapsto \mathsf{e}$, and*
  - *$l(\bigcup_{t \in T_{\mathsf{e}}} (\bigcup_{s \in {}^\bullet t} {}^\bullet s)) \mapsto \mathsf{e}$.*

Proof. The conflict and the enabling relation satisfy Def. 6.6. ■

Example 6.5. *Consider the net N in Fig. 6.1, the associated* DES *$\mathcal{E}_{ss}(N)$ has three events $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$, the conflict relation is empty (not the one of the net, where the transitions t_3, t_4 and t_5 are all pairwise in conflict), and the relation $\mapsto$ is $\{\mathsf{a}, \mathsf{b}\} \mapsto \mathsf{c}$.*

Theorem 6.2. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a semi simple unravel net and $\mathcal{E}_{ss}(N)$ be the associated* DES*. Then w is a trace of N iff w is a trace of $\mathcal{E}_{ss}(N)$.*

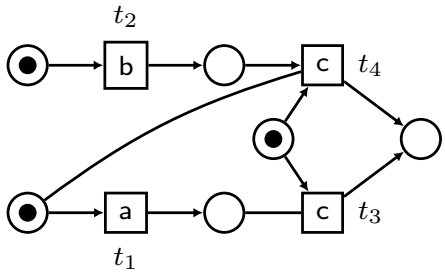
Proof. Since the traces and the configurations are similar of the ones of a BES the proof follows the same pattern of the Th. 3.1. ■

We introduce now the notion of *stable* unravel net. The intuition behind this definition is the following: contextual conditions can be removed (they are initially marked places, and there is no incoming arc connected to these conditions) or they can established and they last *forever* (the places have no outgoing arcs). Given a transition t such that $\underline{t} \neq \varnothing$, with $\mathsf{Modifiers}(t)$ we denote the set of labels associated to the contextual arcs and it is equal to $\{l(s^\bullet) \mid s \in \underline{t}\} \cup \{l({}^\bullet s) \mid s \in \underline{t}\}$, *i.e.* the set of transitions putting or removing tokens in $\underline{t}$.

Definition 6.15. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be an unravel net over the set of labels L. We say that N is* stable *whenever*

1. *C is not empty,*
2. *for all $t \in T$ $\underline{t} \neq \varnothing \;\Rightarrow\; \forall s \in \underline{t}.\, {}^\bullet s = \varnothing \;\vee\; s^\bullet = \varnothing$,*
3. *for all $\mathsf{e} \in l(T)$, for each marking $m \in \mathcal{M}_N$, for all $t, t' \in l^{-1}(\mathsf{e})$ with $t \neq t'$ if $m\,[t\rangle$ then $\neg(m\,[t'\rangle)$, and*
4. *for all $\mathsf{e} \in l(T)$, for all $t, t' \in l^{-1}(\mathsf{e})$ it holds that $\varnothing \neq \underline{t} \neq \underline{t'} \neq \varnothing$ and $\mathsf{Modifiers}(t) = \mathsf{Modifiers}(t')$.*

In stable unravel nets contextual arcs are used to signal modifiers to a label, by allowing to trigger the proper instance of that label (the one corresponding to the proper combination of triggers), whereas in the classical theory of contextual nets ([MR95] or [BCM01]) they are used to either enlarge concurrency or to model asymmetric conflicts. The final conditions are introduced to assure that equally labeled transitions cannot be simultaneously enabled at any marking and have always a non empty context. Stability is not enough to capture shrinking or growing causality. In fact we have to identify, among the various conflicting transitions that are equally labeled which is the one representing the occurrence of the event



N

Figure 6.2: A well formed unravel net with shrinking causality.

without that any modifiers has happened. Once that we have found that transition representing the happening without any modifier, we may find out whether the modifier is a shrinking or growing (or both, but for different labels).

Definition 6.16. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a stable unravel net over the set of labels* L. *We say that N is* well formed *(wfn) iff, for all $\mathsf{e} \in l(T)$, one of the following holds:*

- *if $|l^{-1}(\mathsf{e})| > 1$ then $\exists!\ t \in l^{-1}(\mathsf{e})$ such that $\forall s \in \underline{t}.\ {}^\bullet s = \varnothing$ or*
- *if $l^{-1}(\mathsf{e}) = \{t\}$ then $\underline{t} = \varnothing$.*

This definition guarantees that among the transitions bearing the same label there is one that may happen before that all the modifiers have happened. The second condition say that if there is just one instance of a given label (event) then there is no modifier for that event.

Due to the two definitions above it is possible to introduce the following set of labels. Given a wfn $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ and a label $\mathsf{e} \in l(T)$, with $\mathsf{StableC}(\mathsf{e}) = l(\bigcup_{s \in {}^\bullet t} {}^\bullet s)$, where t is the unique transition in $l^{-1}(\mathsf{e})$ such that either $\forall s \in \underline{t}.\ {}^\bullet s = \varnothing$ or $\underline{t} = \varnothing$, we denote the set of *stable* causes of e. Observe that we are identifying, in the net N, a subset of transitions with a specific label.

Example 6.6. *Consider the stable N net in Fig. 6.2, it is clearly a well formed, the set of modifier for* c *contains just* a *and* $\mathsf{StableC}(\mathsf{c}) = \{(t_2, \mathsf{b})\}$. *Instead of,*

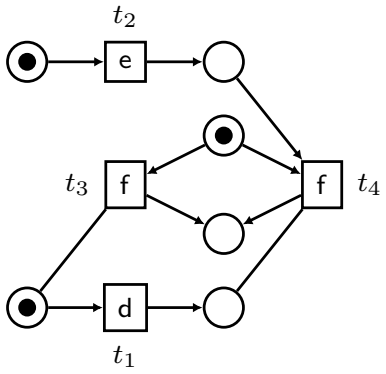


Figure 6.3: A well formed unravel net with growing causality.

considering the **wfn** N' *in Fig. 6.3, the set of modifier for* f *contains just* d *but* $\mathsf{StableC}(\mathsf{f}) = \varnothing$.

The following definition captures what is the usage of contextual arcs in the case of removing causes.

Definition 6.17. *Let* $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ *be a* **wfn** *over the set of labels* L *and let* $\mathsf{e}', \mathsf{e} \in l(T)$ *such that* $\mathsf{e}' \neq \mathsf{e}$. *We say that* e' *is a* shrinking modifier *for* e *iff*

- *there* $\exists t, t' \in l^{-1}(\mathsf{e})$ *such that* $t \neq t'$, $({}^{\diamond}\mathsf{e}', t) \in C$ *and* $(\mathsf{e}'^{\diamond}, t') \in C$ *and*
- *there exists* $\mathsf{e}'' \in \mathsf{StableC}(\mathsf{e})$ *such that* $\mathsf{e}'' \notin l(\bigcup_{s \in {}^{\bullet}t'} {}^{\bullet}s)$.

The set of shrinking modifiers for e *is denoted with* $\mathsf{ShrMod}(\mathsf{e})$.

When the set $\mathsf{ShrMod}(\mathsf{e})$ is not empty, one has to identify what are the dependencies that are dropped. These are captured as follows. Consider $\mathsf{e}' \in \mathsf{ShrMod}(\mathsf{e})$, then there exists $t \in l^{-1}(\mathsf{e})$ such that $({}^{\diamond}\mathsf{e}', t) \in C$. The elements that are dropped are

$$\mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'} = \{l(\bar{t}) \mid \exists ({}^{\diamond}\mathsf{e}', t), (\mathsf{e}'^{\diamond}, t') \in C.\ l(t) = l(t') = \mathsf{e} \ \wedge\ \bar{t} \in {}^{\bullet}({}^{\bullet}t)\} \cap \mathsf{StableC}(\mathsf{e})$$

Similarly we can define the growing modifier, using the contextual conditions as well.

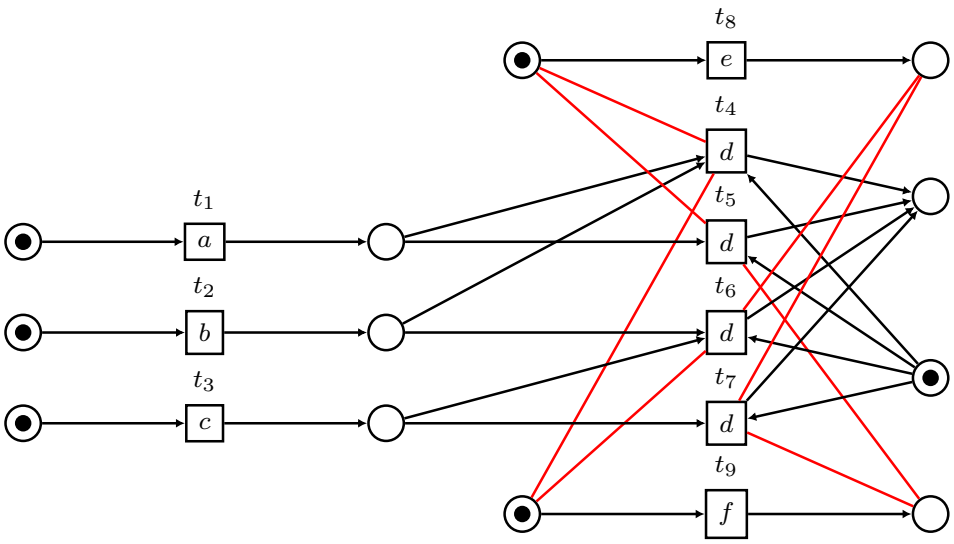


Figure 6.4: A wfn such that $\mathsf{ShrMod}(\mathsf{d}) \neq \varnothing$ and $\mathsf{GroMod}(\mathsf{d}) \neq \varnothing$. Contextual arcs are drawn in red to avoid confusion.

Definition 6.18. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a wfn over the set of labels L and let $\mathsf{e}', \mathsf{e} \in l(T)$ such that $\mathsf{e}' \neq \mathsf{e}$. We say that e' is a* growing modifier *for e iff*

- *there $\exists t, t' \in l^{-1}(\mathsf{e})$ and $\hat{t} \in l^{-1}(\mathsf{e}')$ such that $t \neq t'$, $({}^{\diamond}\mathsf{e}', t) \in C$ and $(\mathsf{e}'^{\diamond}, t') \in C$ and*
- *there exists $\mathsf{e}'' \in l(\bigcup_{s \in {}^{\bullet}t'} {}^{\bullet}s)$ such that $\mathsf{e}'' \notin \mathsf{StableC}(\mathsf{e})$.*

The set of growing modifiers for e is denoted with $\mathsf{GroMod}(\mathsf{e})$.

Again when the set $\mathsf{GroMod}(\mathsf{e})$ is not empty, one has to identify what are the dependencies that are added. Consider again $\mathsf{e}' \in \mathsf{GroMod}(\mathsf{e})$, again there exists $t \in l^{-1}(\mathsf{e})$ such that $({}^{\diamond}\mathsf{e}', t) \in C$. The elements that are added are

$$\mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'} = \{l(t') \mid t' \neq t \ \wedge \ (\mathsf{e}'^{\diamond}, t') \in C\} \backslash l(\mathsf{StableC}(t))$$

Example 6.7. *Consider again the nets of the Fig. 6.2. It has a shrinking modifier a for the label c ($\mathsf{ShrMod}(\mathsf{c}) = \{\mathsf{a}\}$) and $\mathsf{Drop}^{\mathsf{c}}_{\mathsf{a}} = \{\mathsf{b}\}$, whereas the one in the Fig. 6.3 has a growing modifier d for the label f ($\mathsf{GroMod}(\mathsf{f}) = \{\mathsf{d}\}$) and $\mathsf{Add}^{\mathsf{f}}_{\mathsf{d}} = \{\mathsf{e}\}$. In*

the case of the **wfn** *in Fig. 6.4 we have that the label* d *has two modifiers:* e *and* f*, where* e *is both shrinking and growing, whereas* f *is only shrinking.*

The following two propositions characterize event structures associated to wfns where only one kind of modifier is present, either shrinking or growing.

Proposition 6.3. *Let* $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ *be a* **wfn** *such that* $\bigcup_{\mathsf{e}\in l(T)} \mathsf{ShrMod}(\mathsf{e}) \neq \varnothing$ *and* $\bigcup_{\mathsf{e}\in l(T)} \mathsf{GroMod}(\mathsf{e}) = \varnothing$*, then* $\mathcal{E}_{sh}(N) = (l(T), \#, \rightarrow, \rhd)$ *is a* SES*, where*

1. *the conflict relation is defined as follows:* $\mathsf{e} \# \mathsf{e}'$ *iff* $\mathsf{e} \bigstar \mathsf{e}'$*,*
2. *the causality relation is defined as follows: for each* $\mathsf{e} \in l(T)$*, for each* $s \in {}^\bullet l^{-1}(\mathsf{e})$*, for each* $t \in {}^\bullet s$ *we have* $l(t) \rightarrow \mathsf{e}$*, and*
3. *the shrinking causality relation is defined as follows: for all* $\mathsf{e} \in l(T)$*, for each* $\mathsf{e}' \in \mathsf{ShrMod}(\mathsf{e})$ *define* $[\mathsf{e}'' \rightarrow \mathsf{e}] \rhd \mathsf{e}'$ *where* $\mathsf{e}'' \in \mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'}$*.*

Proof. We must prove that 3) implies 2), in particular $\mathsf{e}'' \in \mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'}$ implies $\mathsf{e}'' \rightarrow \mathsf{e}$. Recall $\mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'} = \{l(\bar{t}) \mid \exists({}^\diamond\mathsf{e}', t), (\mathsf{e}'^\diamond, t') \in C.\, l(t) = l(t') = \mathsf{e} \wedge \bar{t} \in {}^\bullet({}^\bullet t)\} \cap \mathsf{StableC}(\mathsf{e})$, it holds $l(\bar{t}) = \mathsf{e}''$ for some $\bar{t} \in {}^\bullet({}^\bullet t)$ and, since e'' is also a stable cause, then it belongs to the set $l(t)$ of item 2), hence $\mathsf{e}'' \rightarrow \mathsf{e}$. ■

Proposition 6.4. *Let* $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ *be a* **wfn** *such that* $\bigcup_{\mathsf{e}\in l(T)} \mathsf{ShrMod}(\mathsf{e}) = \varnothing$ *and* $\bigcup_{\mathsf{e}\in l(T)} \mathsf{GroMod}(\mathsf{e}) \neq \varnothing$*, then* $\mathcal{E}_{gr}(N) = (l(T), \#, \rightarrow, \blacktriangleright)$ *is a* GES*, where*

1. *the conflict relation is defined as follows:* $\mathsf{e} \# \mathsf{e}'$ *iff* $\mathsf{e} \bigstar \mathsf{e}'$*,*
2. *the causality relation is defined as follows: for each* $\mathsf{e} \in l(T)$*, for each* $s \in {}^\bullet l^{-1}(\mathsf{e})$*, for each* $t \in {}^\bullet s$ *we have* $l(t) \rightarrow \mathsf{e}$ *provided that for all* $\mathsf{e}' \in \mathsf{GroMod}(\mathsf{e})$ *it holds that* $l(t) \notin \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'}$*, and*
3. *the growing causality relation is defined as follows: for all* $\mathsf{e} \in l(T)$*, for each* $\mathsf{e}' \in \mathsf{GroMod}(\mathsf{e})$ *define* $\mathsf{e}' \blacktriangleright [\mathsf{e}'' \rightarrow \mathsf{e}]$ *where* $\mathsf{e}'' \in \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'}$*.*

Proof. We prove that $\mathsf{e}' \blacktriangleright [\mathsf{e}'' \rightarrow \mathsf{e}]$ implies $\neg(\mathsf{e}'' \rightarrow \mathsf{e})$. Suppose $\mathsf{e}'' \rightarrow \mathsf{e}$ then $l(t) = \mathsf{e}''$ for each $t \in {}^\bullet s$ and for each $s \in {}^\bullet l^{-1}(\mathsf{e})$ but $l(t) \notin \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'}$ which contradicts the hypothesis on e''. ■

Example 6.8. *The shrinking event structure associate to N in Fig. 6.2 is* $[\mathsf{b} \rightarrow \mathsf{c}]\triangleright\mathsf{a}$*, whereas the growing event structure associated to N' in Fig. 6.3 is* $\mathsf{d} \blacktriangleright [\mathsf{e} \rightarrow \mathsf{f}]$

As before we have the following two theorems.

Theorem 6.3. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a* wfn *such that $\bigcup_{\mathsf{e}\in l(T)} \mathsf{ShrMod}(\mathsf{e}) \neq \varnothing$ and $\bigcup_{\mathsf{e}\in l(T)} \mathsf{GroMod}(\mathsf{e}) = \varnothing$, and let $\mathcal{E}_{sh}(N) = (l(T), \#, \rightarrow, \triangleright)$ be the associated* SES*. Then w is a trace of N iff w is a trace of $\mathcal{E}_{sh}(N)$.*

Proof. ($\Rightarrow$) Let $\sigma \in \mathcal{R}^N_{\mathsf{m}}$ with $\sigma = \mathsf{m}\,[t_1\rangle m_1\,[t_2\rangle m_2 \cdots m_{n-1}\,[t_n\rangle m_n$, and let $\tau = l(t_1 t_2 \cdots t_n)$ the associated trace. We have $l(t_i) \neq l(t_j)$ for all $0 \leqslant i, j \leqslant n$, otherwise $t_i \# t_j$ would be in conflict against the hypothesis of belonging to the same net trace, then the sequence of labels $\tau = l(t_1 t_2 \cdots t_n)$ contains no duplicates, as required by Def. 6.8. Also no labels are in conflict, so the sequence is conflict free. It remains to prove that $\forall 1 \leqslant i \leqslant len(\tau).\ (\mathrm{ic}(\mathsf{e}_i) \setminus \mathrm{dc}(\overline{\tau_{i-1}}, \mathsf{e}_i)) \subseteq \overline{\tau_{i-1}}$. If $len(\tau) = 0$ then the empty sequence of events of $\mathcal{E}_{sh}(N)$ is trivially a trace. Suppose that it is true for net trace of length n. Let $l(t_1 t_2 \cdots t_n t_{n+1})$ the new trace of length $(n+1)$. Let $\mathsf{e} = l^{-1}(t_{n+1})$, we have $lead(\sigma)\,[t_{n+1}\rangle$ where $lead(\sigma)$ is the marking reached executing the firing sequence σ, and since $m_n \subseteq {}^\bullet l^{-1}(t_{n+1})$ and $\{t_n\} \subseteq {}^\bullet m_n$, by Prop. 6.3, $l(t) \rightarrow \mathsf{e}$.

($\Leftarrow$) Again, by induction on the length of the trace. The proof is similar to the ($\Leftarrow$) part of Theorem. 3.1 but we must take into account that a trace of a SES is a sequence of labels and each of these can map into several transitions of the net. However, by condition 2) of Def 6.15 at each marking only one transition bearing the same label can be enabled, so there is no ambiguity in the trace of N. ■

Theorem 6.4. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a* wfn *such that $\bigcup_{\mathsf{e}\in l(T)} \mathsf{ShrMod}(\mathsf{e}) = \varnothing$ and $\bigcup_{\mathsf{e}\in l(T)} \mathsf{GroMod}(\mathsf{e}) \neq \varnothing$, and let $\mathcal{E}_{gr}(N) = (l(T), \#, \rightarrow, \blacktriangleright)$ be the associated*

GES. *Then w is a trace of N iff w is a trace of $\mathcal{E}_{gr}(N)$.*

Proof. ($\Rightarrow$) Let $\sigma \in \mathcal{R}^N_{\mathsf{m}}$ with $\sigma = \mathsf{m}\,[t_1\rangle m_1\,[t_2\rangle m_2 \cdots m_{n-1}\,[t_n\rangle m_n$, and let $\tau = l(t_1 t_2 \cdots t_n)$ the associated trace. We have $l(t_i) \neq l(t_j)$ for all $0 \leqslant i, j \leqslant n$, otherwise $t_i \# t_j$ would be in conflict against the hypothesis of belonging to the same net trace, then the sequence of labels $\tau = l(t_1 t_2 \cdots t_n)$ contains no duplicates, as required by Def. 6.8. Also no labels are in conflict, so the sequence is conflict free. It remains to prove that $\forall 1 \leqslant i \leqslant \mathit{len}(\tau).\ (\mathrm{ic}(\mathsf{e}_i) \cup \mathrm{ac}(\overline{\tau_{i-1}}, \mathsf{e}_i)) \subseteq \overline{\tau_{i-1}}$. If $\mathit{len}(\tau) = 0$ then the empty sequence of events of $\mathcal{E}_{sh}(N)$ is trivially a trace. Suppose that it is true for net trace of length n. Let $l(t_1 t_2 \cdots t_n t_{n+1})$ the new trace of length $(n+1)$. Let $\mathsf{e} = l^{-1}(t_{n+1})$, we have $\mathit{lead}(\sigma)\,[t_{n+1}\rangle$ where $\mathit{lead}(\sigma)$ is the marking reached executing the firing sequence σ, and since $m_n \subseteq {}^\bullet l^{-1}(t_{n+1})$ and $\{t_n\} \subseteq {}^\bullet m_n$, by Prop. 6.4, $l(t) \rightarrow \mathsf{e}$ because it is a stable cause of e, *i.e.* cannot belong to $\mathsf{Add}^{l(t_{n+1})}_{l(t)}$.

($\Leftarrow$) Again, by induction on the length of the trace. The proof is similar to the ($\Leftarrow$) part of Theorem. 3.1 but we must take into account that a trace of a SES is a sequence of labels and each of these can map into several transitions of the net. However, by condition 2) of Def 6.15 at each marking only one transition bearing the same label can be enabled, so there is no ambiguity in the trace of N. ■

We can now put together what we have seen up to now obtaining a DCES.

Proposition 6.5. *Let $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ be a **wfn** such that if $\mathsf{e}' \in \big(\bigcup_{\mathsf{e}\in l(T)} \mathsf{ShrMod}(\mathsf{e}) \cap \bigcup_{\mathsf{e}\in l(T)} \mathsf{GroMod}(\mathsf{e}) \neq \varnothing\big)$ it holds that $\mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'} \cap \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'} = \varnothing$, then $\mathcal{E}_{sg}(N) = (l(T), \#, \rightarrow, \rhd, \blacktriangleright)$ is a* DCES*, where*

- *the conflict relation is defined as follows: $\mathsf{e} \# \mathsf{e}'$ iff $\mathsf{e} \bigstar \mathsf{e}'$,*
- *the causality relation is defined as follows: for each $\mathsf{e} \in l(T)$, for each $s \in {}^\bullet l^{-1}(\mathsf{e})$, for each $t \in {}^\bullet s$ we have $l(t) \rightarrow \mathsf{e}$ provided that for all $\mathsf{e}' \in \mathsf{GroMod}(\mathsf{e})$ it holds that $l(t) \notin \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'}$,*
- *the shrinking causality relation is defined as follows: for all $\mathsf{e} \in l(T)$, for each $\mathsf{e}' \in \mathsf{ShrMod}(\mathsf{e})$ define $[\mathsf{e}'' \rightarrow \mathsf{e}] \rhd \mathsf{e}'$ where $\mathsf{e}'' \in \mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'}$, and*

- *the growing causality relation is defined as follows: for all* $\mathsf{e} \in l(T)$*, for each* $\mathsf{e}' \in \mathsf{GroMod}(\mathsf{e})$ *define* $\mathsf{e}'' \blacktriangleright [\mathsf{e} \to \mathsf{e}']$ *where* $\mathsf{e}'' \in \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'}$.

Proof. It follows from Prop. 6.3 and 6.4. ■

Example 6.9. *Consider the net in Fig. 6.4. The sets of modifiers for* d *are* $\mathsf{ShrMod}(\mathsf{d}) = \{\mathsf{e}, \mathsf{f}\}$ *and* $\mathsf{GroMod}(\mathsf{d}) = \{\mathsf{e}\}$, $\mathsf{Drop}^{\mathsf{d}}_{\mathsf{e}} = \{\mathsf{a}\}, \mathsf{Drop}^{\mathsf{d}}_{\mathsf{f}} = \{\mathsf{b}\}$ *and* $\mathsf{Add}^{\mathsf{d}}_{\mathsf{e}} = \{\mathsf{c}\}$ *and finally* $\mathsf{StableC}(\mathsf{d}) = \varnothing$. *The associated* DCES *has the following enabling relation:* $\mathsf{a} \to \mathsf{d}$ *and* $\mathsf{b} \to \mathsf{d}$*, the conflict relation is empty, the shrinking causality relation is* $[\mathsf{a} \to \mathsf{d}] \rhd \mathsf{e}$ *and* $[\mathsf{b} \to \mathsf{d}] \rhd \mathsf{f}$ *and the growing causality relation is* $\mathsf{e} \blacktriangleright [\mathsf{c} \to \mathsf{d}]$.

Theorem 6.5. *Let* $N = \langle S, T, F, C, \mathsf{m}, l\rangle$ *be a* ***wfn*** *such that if* $\mathsf{e}' \in \big(\bigcup_{\mathsf{e}\in l(T)} \mathsf{ShrMod}(\mathsf{e}) \cap \bigcup_{\mathsf{e}\in l(T)} \mathsf{GroMod}(\mathsf{e}) \neq \varnothing\big)$ *it holds that* $\mathsf{Drop}^{\mathsf{e}}_{\mathsf{e}'} \cap \mathsf{Add}^{\mathsf{e}}_{\mathsf{e}'} = \varnothing$*, and let* $\mathcal{E}_{sg}(N) = (l(T), \#, \to, \rhd, \blacktriangleright)$ *be the associated* DCES. *Then* w *is a trace of* N *iff* w *is a trace of* $\mathcal{E}_{sg}(N)$.

Proof. It follows from Th. 6.3 and 6.4. ■

6.4.2 From DCES to unravel nets

The idea here to associate to each event of the event structure a set of equally labeled transitions, each of them represent one of the possible *enabling conditions* of the event.

We start defining a number of sets we will use in the following. Consider a DCES $\Sigma = (\mathsf{E}, \#, \to, \rhd, \blacktriangleright)$ then, for each $\mathsf{e} \in \mathsf{E}$

- $\mathsf{Before}(\mathsf{e}) = \{\mathsf{e}' \mid \mathsf{e}' \to \mathsf{e} \ \vee \ \mathsf{e}'' \blacktriangleright [\mathsf{e}' \to \mathsf{e}]\}$ is the set of events that causally are before e, also *potentially*, in the case of growing causality,

- $\mathsf{After}(\mathsf{e}) = \{\mathsf{e}' \mid \mathsf{e} \to \mathsf{e}' \ \vee \ \mathsf{e}'' \blacktriangleright [\mathsf{e} \to \mathsf{e}']\}$ is the set of events that causally are after e, also *potentially*, in the case of growing causality,

- $\mathsf{ModShr}(\mathsf{e}) = \{\mathsf{e}' \mid [\mathsf{e}'' \to \mathsf{e}] \rhd \mathsf{e}'\}$ is the set of shrinking modifiers for a certain event e,
- $\mathsf{ModGro}(\mathsf{e}) = \{\mathsf{e}' \mid \mathsf{e}' \blacktriangleright [\mathsf{e}'' \to \mathsf{e}]\}$ is the set of shrinking modifiers for a certain event e,
- $\mathsf{Less}(\mathsf{e}, X) = \{\mathsf{e}' \mid \exists \mathsf{e}'' \in X.\ [\mathsf{e}' \to \mathsf{e}] \rhd \mathsf{e}''\}$ is the set of cause of e that may be removed due to a modifier in the set X,
- $\mathsf{More}(\mathsf{e}, X) = \{\mathsf{e}' \mid \exists \mathsf{e}'' \in X.\ \mathsf{e}'' \blacktriangleright [\mathsf{e}' \to \mathsf{e}]\}$ is the set of cause of e that may be added due to a modifier in the set X, and
- $\mathsf{Solid}(\mathsf{e}) = \{\mathsf{e}' \mid \mathsf{e}' \to \mathsf{e}\}$.

For each event e and each subset of modifiers $\mathsf{ModShr}(\mathsf{e}) \cup \mathsf{ModGro}(\mathsf{e})$ we will introduce a transition labeled with e. Among all the transitions with the same labels the one that should be executed is the one where all and only the modifiers in the subset have been previously executed.

Proposition 6.6. *Let $\Sigma = (\mathsf{E}, \#, \to, \rhd, \blacktriangleright)$ be a* DCES. *Then to Σ we associate the wfn $\mathcal{N}(\Sigma) = \langle S, T, F, C, \mathsf{m}, l\rangle$ where*

1. $S = (\mathsf{E}\times\{*\}) \cup (\mathsf{E}\times\{\circ\}) \cup \{(\mathsf{e}, \mathsf{e}', \to) \mid \mathsf{e}' \in \mathsf{After}(\mathsf{e})\} \cup \{(\mathsf{e}, \mathsf{e}', \to) \mid \mathsf{e} \in \mathsf{Before}(\mathsf{e}')\} \cup \{(\{\mathsf{e}, \mathsf{e}'\}, \#) \mid \mathsf{e} \# \mathsf{e}'\}$,
2. $T = \{(\mathsf{e}, X) \mid \mathsf{e} \in \mathsf{E} \ \wedge \ X \subseteq \mathsf{ModShr}(\mathsf{e}) \cup \mathsf{ModGro}(\mathsf{e})\}$,
3. $(s, t) \in F$ *whenever one the following holds:*
   (a) $s = (\mathsf{e}, *)$ *and* $t = (\mathsf{e}, X)$, *for any* X,
   (b) $s = (\mathsf{e}', \mathsf{e}, \to)$ *and* $t = (\mathsf{e}, X)$ *with either* $\mathsf{e}' \in \mathsf{Solid}(\mathsf{e}) \backslash \mathsf{Less}(\mathsf{e}, X \cap \mathsf{ModShr}(\mathsf{e}))$ *or* $\mathsf{e}' \in \mathsf{Solid}(\mathsf{e}) \backslash \mathsf{More}(\mathsf{e}, X \cap \mathsf{ModGro}(\mathsf{e}))$,
   (c) $s = (Y, \#)$, *with* $t = (\mathsf{e}, X)$ *and* $\mathsf{e} \in Y$, *for any* X,
4. $(t, s) \in F$ *whenever one the following holds:*

(a) $s = (\mathsf{e}, \circ)$ *and* $t = (\mathsf{e}, X)$*, for any* X*,*

(b) $s = (\mathsf{e}, \mathsf{e}', \rightarrow)$ *and* $t = (\mathsf{e}, X)$*, for any* X*,*

5. $(s, t) \in C$ *whenever*

(a) $s = (\mathsf{e}', *)$ *and* $t = (\mathsf{e}, \varnothing)$*, for all* $\mathsf{e}' \in \mathsf{ModGro}(\mathsf{e}) \cup \mathsf{ModShr}(\mathsf{e}) \neq \varnothing$*,*

(b) $s = (\mathsf{e}', *)$ *and* $t = (\mathsf{e}, X)$ *if* $\mathsf{e}' \notin X \cap \mathsf{ModShr}(\mathsf{e})$*,*

(c) $s = (\mathsf{e}', \circ)$ *and* $t = (\mathsf{e}, X)$*, if* $\mathsf{e}' \in X \cap \mathsf{ModGro}(\mathsf{e})$*,*

6. $\mathsf{m}(s) = 1$ *iff* s *is equal to* $(\mathsf{e}, *)$ *or to* $(\mathsf{e}, \mathsf{e}', \#)$ *and it is equal to* 0 *otherwise, and*

7. $l(\mathsf{e}, X) = \mathsf{e}$.

Proof. We first prove that $\mathcal{N}(\Sigma)$ is stable (see 6.15). Requirements 1) and 2) of Def. 6.15 are trivially achieved by 5). To prove 3) of 6.15 we notice the each transition with the same label e, (e, X), differs by the subsets X of modifiers which has been executed before e, this means that when one of transitions of $l^{-1}(\mathsf{e})$ is executed the preset of the others bearing the same label is partially marked because the subset of modifier (which define the preset) is different (otherwise it would be the same transition). Last requirement for being a stable unravel net says that all transitions with the same label have a not empty context and the modifiers map in the same sets of labels. This is true in $\mathcal{N}(\Sigma)$ because, by construction all transition are built taking all possible subsets of modifiers of the same label.

Condition 5) captures the fact that $\mathcal{N}(\Sigma)$ is a wfn (Def. 6.16). ■

Each target of modifiers has a number of different *implementations* depending on the set of modifiers that has happened. The flow and contextual relation are defined easily using the sets of labels defined before Prop. 6.6.

Example 6.10. *Consider the* DCES *where the causal relation is* $\mathsf{a} \rightarrow \mathsf{d}$ *and* $\mathsf{b} \rightarrow \mathsf{d}$*, the empty conflict relation, the shrinking causality relation is* $[\mathsf{a} \rightarrow \mathsf{d}] \rhd \mathsf{e}$ *and*

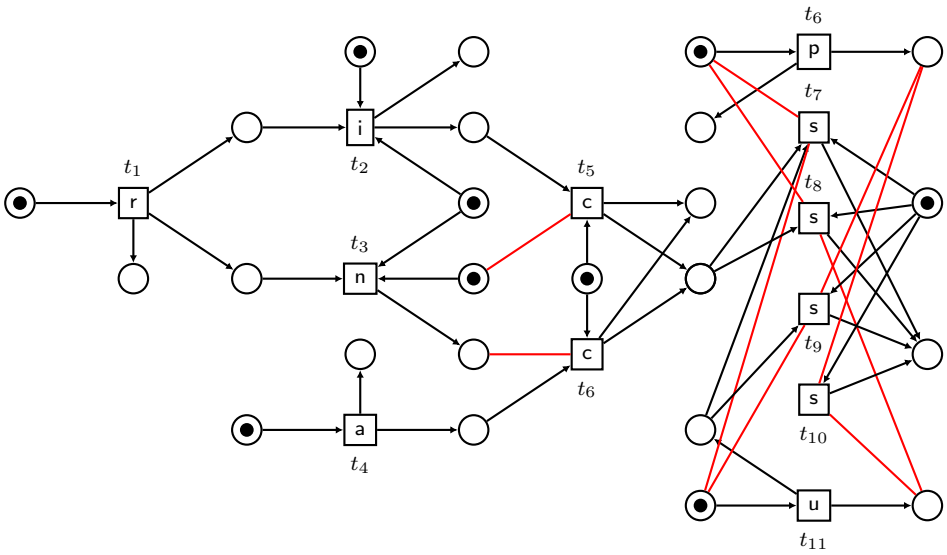


Figure 6.5: The unravel net of the seller in example 6.4

$[\mathsf{b} \to \mathsf{d}] \rhd \mathsf{f}$ *and the growing causality relation is* $\mathsf{e} \blacktriangleright [\mathsf{c} \to \mathsf{d}]$. *It is straightforward to check that the synthesized* wfn *is the net in Fig. 6.4.*

Theorem 6.6. *Let* $\Sigma = (\mathsf{E}, \#, \to, \rhd, \blacktriangleright)$ *be a* DCES, *and let* $\mathcal{N}(\Sigma) = \langle S, T, F, C, \mathsf{m}, l \rangle$ *be the associated unravel net. Then* w *is a trace of* Σ *iff* w *is a trace of* $\mathcal{N}(\Sigma)$.

Proof. ($\Rightarrow$) By induction on the length of a trace w.

If w is the empty trace, then w is also a trace of $\mathcal{N}(\Sigma)$. If $len(w) = 1$ and $\overline{w} = \{\mathsf{e}\}$ then $l^{-1}(\mathsf{e})$ is connected to the initial marking, *i.e.* ${}^\bullet l^{-1}(\mathsf{e}) \subseteq S_m$, hence e is also a trace of $\mathcal{N}(\Sigma)$. Assume that, for $len(w) = n$, it holds $w \in \mathsf{Tr}(\mathcal{N}(\Sigma))$, $w = \mathsf{e}_1 \cdots \mathsf{e}_n$ with $\overline{w} \subseteq \mathsf{E}$ and consider the trace $w\mathsf{e}_{n+1} \in \mathsf{Tr}(\Sigma)$. According to Def. 6.12 we can have four cases:

1. $\mathrm{ac}(\overline{w_n}, \mathsf{e}_{n+1})) = \mathrm{dc}(\overline{w_n}, \mathsf{e}_{n+1})) = \varnothing$.

   Then $\mathrm{ic}(\mathsf{e}_{n+1}) \subseteq \overline{w_n}$. Hence when $l^{-1}(\mathsf{e}_n)$ is executed all places of the form $(\mathsf{e}', \mathsf{e}_{n+1}, \to)$ are marked which coincide with ${}^\bullet(\mathsf{e}_{n+1}, \varnothing)$,

2. $\mathrm{dc}(\overline{w_n}, \mathsf{e}_{n+1})) = \varnothing$.

   The causes of e_{n+1} are all the initial ones plus those added by the adding modifiers, which lead to one of the presets of the transitions $l^{-1}(\mathsf{e}_{n+1})$ to be

satisfied (not more than one because it is a stable unravel net),

3. $\mathrm{ac}(\overline{w_n}, \mathsf{e}_{n+1})) = \varnothing$ or $\varnothing \neq \mathrm{ac}(\overline{w_n}, \mathsf{e}_{n+1})) \neq \mathrm{dc}(\overline{w_n}, \mathsf{e}_{n+1})) \neq \varnothing$. This case can be treated as 2), *i.e.* when $l^{-1}(\mathsf{e}_n)$ is executed (and only one of them is) only one of the preset of $l^{-1}(\mathsf{e}_{n+1})$ will be marked.

($\Leftarrow$) Again we do the proof by induction on the length of the trace. For the empty trace the thesis is trivial. Let $u = l(t_1 t_2 \cdots t_n) \in \mathsf{Tr}(\mathcal{N}(\Sigma))$. Then there exists a fs σ such that $run(\sigma) = u$. Consider the trace $u \cdot t_{n+1} \in \mathsf{Tr}(\mathcal{N}(\Sigma))$. By inductive hypothesis $l(t_1 t_2 \cdots t_n)$ is a trace of Σ, and that t_{n+1} is enabled at the marking $lead(\sigma)$. We show that $((\mathrm{ic}(l(t_{n+1})) \cup \mathrm{ac}(\overline{l(u_n)}, l(t_{n+1})) \setminus \mathrm{dc}(\overline{l(u_n)}, l(t_{n+1}))) \subseteq \overline{u_n}$.

As t_{n+1} is enabled at the marking $lead(\sigma)$, we have that the places $(e', e, \rightarrow)$ are marked, where $\mathsf{e} = l(t_{n+1})$ and each e' satisfies $((\mathrm{ic}(\mathsf{e}') \cup \mathrm{ac}(\overline{l(u_{n-1})}, \mathsf{e}') \setminus \mathrm{dc}(\overline{l(u_{n-1})}, \mathsf{e}')) \subseteq \overline{u_{n-1}}$. Then

1. $\mathsf{e}' \in \overline{u_n}$ for each e',

2. $\mathsf{e}' \in ((\mathrm{ic}(l(t_{n+1})) \cup \mathrm{ac}(\overline{l(u_n)}, l(t_{n+1})) \setminus \mathrm{dc}(\overline{l(u_n)}, l(t_{n+1})))$ for each e',

From these two conditions it follows that ($\overline{u_{n-1}} \cap$ all the e' defined above) $\subseteq \overline{u_n}$. Clearly $\overline{u\mathsf{e}}$ is conflict free. ■

Observe that $\mathcal{E}_{sg}(\mathcal{N}(\Sigma))$ gives indeed Σ but this is not in general true if, starting from a wfn N satisfying all the condition in Prop. 6.5 we do not in general obtain the same net after applying $\mathcal{N}(\mathcal{E}_{sg}(N))$.

Example 6.11. *Consider the seller described in the introduction and formalized as* DCES Σ *in Ex. 6.4. The associated net* $\mathcal{N}(\Sigma)$ *is the one depicted in Fig. 6.5. The event* s *has four instances, as it has two modifiers (*p *and* u*), hence the four transitions are* $(\mathsf{s}, \varnothing), (\mathsf{s}, \{\mathsf{p}\}), (\mathsf{s}, \{\mathsf{u}\})$ *and* $(\mathsf{s}, \{\mathsf{p}, \mathsf{u}\})$*, whereas the event* c *has just two instances as there is just a modifier acting as a modifier letting the dependencies grow and shrink at the same time.*

Chapter 7

Conclusion

We try to draw some conclusions that are more a way to settle what we have reached and what we can imagine to do in the future development of the ideas we have presented.

The unravel nets turned out to be a flexible tool for representing in a compact way the non-sequential behavior of Petri nets. Furthermore, as in the case of causal nets, they have a strong connection with the event structures domain.

We have shown that bundle event structures can model faithfully the behaviours of unravel nets and vice versa, *w.r.t* event traces. Then they have been put into the context of the classic non-sequential semantics for Petri nets, namely the one of unfoldings, and they proved to capture their compact representations. From this we developed a general framework which, through the notion of *merging relation*, is capable of defining different criteria for merging places and transitions. Finally we took a more expressive notion of event structure and shown that the unravel nets with read arcs are the proper counterpart of some of these notions.

We foresee some developments that we briefly summarize.

7.1 Causal conditions

The notion of unravel net allows one to obtain causal dependencies that have a local flavour which, in the context of property verification, is of limited use. It would instead be nice to be able to add some conditions to ensure that the whole subnet identified by these conditions is acyclic.

An attempt in this direction is the one proposed in [CP14] where a notion of *causal condition* is proposed. Roughly speaking a causal condition is associated to the $i-th$ resource with a certain label (hence, if the resource is b, $\mathsf{tok}(b) = i$) and it records all the possible ways this token is produced. This information is based on the notion of *neighbourhood* of a transition. Formally the neighbourhood is defined as follows: Let $N = \langle S, T, F, \mathsf{m}\rangle$ be a net, and let $t \in T$, with $\circlearrowleft(t) = \{t' \mid t' \in {}^\bullet s \text{ or } t' \in s'^\bullet \text{ with } s \in {}^\bullet t \text{ and } s' \in t^\bullet\} \cup \{t\}$ we denote the *neighborhood* of t, namely the transitions *following* and *preceding* t, including t. The transitions in the neighborhood of t are used to find the *local name* of the occurrence of the transition in the unfolding, and the name is used to characterize also internal and control places. To this aim, we introduce an equivalence on words (on alphabets containing the names of transitions in the net we have to unfold). Let $N = (\langle S, T, F, \mathsf{m}\rangle, \nu)$ be a multi-clock net, and let $T' \subseteq T$ be a subset of transitions, and let w, w' two words on $(T')^+$, the for all $t \in T'$, we say that $w \sim_t w'$ iff for all $s \in [\![{}^\bullet t]\!]$, $|proj(w, {}^\bullet s)| = |proj(w', {}^\bullet s)|$ and for all $s \in [\![t^\bullet]\!]$, $|proj(w, {}^\bullet s)| = |proj(w', {}^\bullet s)|$, where the operation $proj(w, X)$ takes a word on an alphabet containing X and deletes all the symbols that are not in X and with $(\!|w|\!)_{\sim_t}$ we denote the equivalence class of the word w. Control places and transitions are pairs where the first component is an equivalence class of words on an alphabet of transitions (restricted to the transitions of an automata forming the multi-clock net) and the second component is a transition. The first component of a control place, the equivalence class, encodes all the equivalent (local) histories

leading to the same future, represented by the name of the transition in the second component.

Unfortunately adding these conditions (connected to the events consuming the resource they are associated to, or to the events producing that resource and that are in the one of the local histories coded by the word) does not enforce acyclicity when restricting to these places, for the same reason it does not in mergend processes. Furthermore control places do not guarantee that the resulting net is an unravel one.

Still it is worth investigating if a suitable subset of control places and conflict places can lead to positive results.

7.2 Heterogeneous partitions

Merging criteria depends on which kind of information on places we consider to be of any interest. For instance, merged processes take the token occurrences while trellis processes take time occurrences of a labeled place. We can think of combining those two or defining new ones, and, with or without conflict places we are able to maintain the unravel net status. If we are able to *partition* unravel nets, one could imagine to apply different criteria to different partitions. For example, consider a net with three partitions ν_1, ν_2, ν_3, where ν_1 is a generic branching processes, ν_2 is a branching processes result of the unfolding of a multi-clock net, and ν_3 is an unravel net with no specific characteristics. It would be interesting to study how different merging criteria behave in such scenario: *e.g.* if we use occurrence depth in ν_1, height in ν_2, and another proper measure in ν_3 will the *unravelness* property preserved? if not, it is possible to enforce it? How do this merging would perform with respect a suitable *global* merging relation? We do believe that the answer to these questions is positive.

7.3 Merging relation and contextual unravel nets

The kind of contextual unravel nets we have considered can be easily reduced to unravel nets without read arcs. One may think to develop a way to define merging relations to include contexts. In [RSK13] the authors introduce merged processes for contextual Petri nets (CMP). From our point of view several questions arise:

- Are contextual merged processes also contextual unravel nets?
- If the answer to the previous question is negative, can they the property of being an unravel net somehow forced as we did in this thesis?
- Since CMP configurations differ from classic mp-configurations, can we find the same correspondence with configurations of contextual unravel nets?

7.4 Reveals relations vs merging relation

Reveals relations [BCH13, HKS13] are relations on transitions defined in causal nets. A transition a reveals another transition b, $a \rhd b$ if for any maximal run containing a, b is present. Reveal relations are neither symmetric nor antisymmetric, but their symmetric closure turns them into equivalence relations. The set of transitions partitioned in this way, called *facets*, are such that if one belongs to a run, then all the equivalent transitions in the class are in it, which implies that runs can be seen as set of facets. By quotienting the net under these relations one obtain a more concise representation of the causal net called *tight* net. Since causal nets are unravel nets, it is natural to think if the tight nets are and, if so, what relationships can be find between reveal and merging relations.

Bibliography

[AKPN15a] Youssef Arbach, David Karcher, Kirstin Peters, and Uwe Nestmann. Dynamic causality in event structures. In Susanne Graf and Mahesh Viswanathan, editors, *FORTE 2015*, volume 9039 of *Lecture Notes in Computer Science*, pages 83–97. Springer, 2015.

[AKPN15b] Youssef Arbach, David Karcher, Kirstin Peters, and Uwe Nestmann. Dynamic causality in event structures (technical report). *CoRR*, abs/1504.00512, 2015.

[Bal00] Paolo Baldan. *Modelling concurrent computations: from contextual Petri nets to graph grammars.* PhD thesis, Department of Computer Science, University of Pisa, 2000. Available as technical report n. TD-1/00.

[BB87] Tommaso Bolognesi and Ed Brinksma. Introduction to the ISO specification language LOTOS. *Computer Networks*, 14:25–59, 1987.

[BBCP04] Paolo Baldan, Nadia Busi, Andrea Corradini, and G. Michele Pinna. Domain and event structure semantics for Petri nets with read and inhibitor arcs. *Theoretical Computer Science*, 323(1-3):129–189, 2004.

[BCH13] Sandie Balaguer, Thomas Chatain, and Stefan Haar. Building occurrence nets from reveals relations. *Fundamamenta Informaticae*, 123(3):245–272, 2013.

[BCM01] Paolo Baldan, Andrea Corradini, and Ugo Montanari. Contextual Petri nets, asymmetric event structures and processes. *Information and Computation*, 171(1):1–49, 2001.

[Bou90] Gérard Boudol. Flow Event Structures and Flow Nets. In Irène Guessarian, editor, *Semantics of Systems of Concurrent Processes*, volume 469 of *Lecture Notes in Computer Science*, pages 62–95. Springer, 1990.

[CP14] Giovanni Casu and G. Michele Pinna. Flow unfolding of multi-clock nets. In Gianfranco Ciardo and Ekkart Kindler, editors, *Petri Nets 2014*, volume 8489 of *Lecture Notes in Computer Science*, pages 170–189. Springer, 2014.

[CP16] Giovanni Casu and G. Michele Pinna. An unifying framework for compacting petri nets behaviors. In Vittorio Bilò and Antonio Caruso, editors, *Proceedings of ICTCS 2016*, 2016. to appear.

[CP17a] Giovanni Casu and G. Michele Pinna. Merging relations: a way to compact petri nets' behaviors uniformly. In Carlos M. Vidé, editor, *Proceeding of LATA 2017*, Lecture Notes in Computer Science. Springer, 2017. to appear.

[CP17b] Giovanni Casu and G. Michele Pinna. Petri nets and dynamic causality for service-oriented computations. In *Proceedings of SAC 2017*. ACM, 2017. to appear.

[DR15] Jörg Desel and Wolfgang Reisig. The concepts of Petri nets. *Software and System Modeling*, 14(2):669–683, 2015.

[Eng91] Joost Engelfriet. Branching processes of Petri nets. *Acta Informatica*, 28(6):575–591, 1991.

[ERV02] Javier Esparza, Stefan Römer, and Walter Vogler. An Improvement of McMillan's Unfolding Algorithm. *Formal Methods in System Design*, 20(3):285–310, 2002.

[Fab07] Eric Fabre. Trellis processes : A compact representation for runs of concurrent systems. *Discrete Event Dynamic Systems*, 17(3):267–306, 2007.

[Gun92] Jeremy Gunawardena. Causal automata. *Theoretical Computer Science*, 101(2):265–288, 1992.

[HKS13] Stefan Haar, Christian Kern, and Stefan Schwoon. Computing the reveals relation in occurrence nets. *Theoretical Computer Science*, 493:66–79, 2013.

[Kat96] Joost-Pieter Katoen. *Quantitative and Qualitative Extensions of Event Structures*. PhD thesis, Enschede: Centre for Telematics and Information Technology, 1996.

[KKKV06] Victor Khomenko, Alex Kondratyev, Maciej Koutny, and Walter Vogler. Merged Processes: a new condensed representation of Petri net behaviour. *Acta Informatica*, 43(5):307–330, 2006.

[Lan92] Rom Langerak. Bundle Event Structures: A Non-Interleaving Semantics for Lotos. In Michel Diaz and Roland Groz, editors, *FORTE '92*, volume C-10 of *IFIP Transactions*, pages 331–346. North-Holland, 1992.

[Lan93] Rom Langerak. Bundle event structures: A non-interleaving semantics for lotos. In Michel Diaz and Roland Groz, editors, *FORTE '92*, volume C-10 of *IFIP Transactions*, pages 331–346. North-Holland, 1993.

[McM93] Kenneth L. McMillan. Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits. In *CAV '92*, LNCS 663, pages 164–177, 1993.

[MR95] Ugo Montanari and Francesca Rossi. Contextual nets. *Acta Informatica*, 32(6), 1995.

[NPW81] Mogens Nielsen, Gordon Plotkin, and Glynn Winskel. Petri Nets, Event Structures and Domains, Part 1. *Theoretical Computer Science*, 13:85–108, 1981.

[Pet66] Carl Adam Petri. *Communication with automata.* PhD thesis, Universität Hamburg, 1966.

[PP95] G. Michele Pinna and Axel Poigné. On the nature of events: another perspective in concurrency. *Theoretical Computer Science*, 138(2):425–454, 1995.

[Rei85] Wolfgang Reisig. *Petri Nets: An Introduction.* EACTS Monographs on Theoretical Computer Science. Springer Verlag, 1985.

[Rei13] Wolfgang Reisig. *Understanding Petri Nets - Modeling Techniques, Analysis Methods, Case Studies.* Springer, 2013.

[RSK13] César Rodríguez, Stefan Schwoon, and Victor Khomenko. Contextual merged processes. In *Application and Theory of Petri Nets and Concurrency - 34th International Conference, PETRI NETS 2013, Milan, Italy, June 24-28, 2013. Proceedings*, pages 29–48, 2013.

[vP09] Rob J. van Glabbeek and Gordon Plotkin. Configuration structures, event structures and Petri nets. *Theoretical Computer Science*, 410(41):4111–4159, 2009.

[Win87] Glynn Winskel. Event Structures. In Wilfried Brauer, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Petri Nets: Central Models and Their Properties*, volume 255 of *Lecture Notes in Computer Science*, pages 325–392. Springer, 1987.

[Win88] Glynn Winskel. An introduction to event structures. In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, School/Workshop, Noordwijkerhout, The Netherlands, May 30 - June 3, 1988, Proceedings*, pages 364–397, 1988.