Removing Confusion in Locally Finite Petri Nets

Author
Terts Diepraam
Amsterdam University College
terts.diepraam@gmail.com

Supervisor
Dr. Lorenzo Galeotti
Amsterdam University College
lorenzo.galeotti@gmail.com

Tutor
Dr. Cor Zonneveld
Amsterdam University College
c.zonneveld@auc.nl

Reader
Dr. Yurii Khomskii
Amsterdam University College
Institute for Logic, Language and Computation
yurii@deds.nl

Major: Sciences

June 3, 2020

9954 words

Abstract

This thesis aims to explore methods for randomising Petri nets. We focus on the s-cell method by Bruni, Melgratti, and Montanari [5] and the branching cells method by Abbes and Benveniste [2] that apply to Petri nets with confusion. Additionally, we implement the s-cell method in a Python program, which can be found at `https://gitlab.com/tertsdiepraam/petrinet`. To do so, we redefine the $\ominus$ operation and formalise the pruning operations from [5]. The redefined $\ominus$ operation might also provide a small first step towards applying the s-cell method to locally finite Petri nets.

Keywords — true-concurrency, Petri nets, event structures, confusion, structural branching cells

Contents

Chapter 1

Introduction

Concurrency theory gained popularity due to the rise of parallelisation in computing and long-distance communication methods. A process is called *concurrent* if the order of some events can be changed without affecting the final state of the process. Since concurrent events can often be executed in parallel, concurrency is often used to optimise computer programs. However, the concept of concurrency applies to more than just computer science. In fact, concurrency is not an exceptional property; many processes such as chemical reactions and traffic are concurrent.

Models of concurrency can be distinguished based on whether they use *interleaving* or *true-concurrent* semantics. In a true-concurrent model, the events are only partially ordered. In an interleaving model, the events are totally ordered. In this paper, we focus on two models of concurrency: Petri nets and event structures.

Probability is often used in models for concurrent processes to make the choices probabilistic which is, for example, useful for simulation and performance analysis of a process. Active research is being done to investigate how probabilistic choices can be added to Petri nets. Ideally, such a model satisfies the following conditions:

1. the behaviour of the probabilistic model must match the behaviour of standard Petri nets;
2. equivalent processes must have equal probabilities;
3. the sum of probabilities of all maximal processes should be 1; and
4. the probabilities of executing concurrent events should be independent.

Satisfying the third condition is especially difficult, since it is possible to construct a system where concurrent events are dependent on each other. When this is the case, we say that a system has *confusion*. Otherwise, we say that it is *confusion-free*.

Stochastic Petri nets are a probabilistic model that uses interleaving semantics [4]. However, this model significantly changes the behaviour of Petri nets. Hence, this model violates the first condition above [4, p. 6]. A probabilistic model for concurrency using true-concurrent semantics was introduced by Varacca, Völzer, and Winskel [12]. This model satisfies all conditions above, but only applies to confusion-free Petri nets. Two other methods for attaching probabilities to Petri nets with confusion using true-concurrent semantics have been developed: the *branching cells* method was introduced by Abbes and Benveniste [2] and the *structural branching cells* or *s-cells* method by Bruni, Melgratti, and Montanari [5]. The first is a dynamic method that removes confusion by controlling the execution of the Petri net. The second removes confusion by changing the Petri net statically to a confusion-free net. However, the method by Bruni, Melgratti, and Montanari [5] applies only to acyclic Petri nets, while the method by Abbes and Benveniste [2] applies to locally finite Petri nets, a subclass of Petri nets which allows for "controlled" cycles.

In this paper, we explore the branching cells method and the s-cells method. Additionally, we have implemented the s-cells method in a Python program, which required further formalisation of the pruning step of the s-cells method.

This capstone is structured as follows: Chapter 2 treats the mathematical preliminaries in set theory and probability theory. Chapter 3 details the classes of models for concurrent processes. Chapters 4 and 5 explain two of these models, Petri nets and event structures, in depth. Finally, Chapters 6 to 8 explore the methods for attaching probabilities to these models by Varacca, Völzer, and Winskel [12], Abbes and Benveniste [2] and Bruni, Melgratti, and Montanari [5], respectively.

Chapter 2

Preliminaries

In this chapter we will introduce the mathematical notions that are used throughout this paper.

2.1 Relations, Orders and Graphs

For binary relations, we introduce the transitive closure in the standard way.

Definition 2.1 (Transitive closure)**.** Let $F \subseteq A \times A$ be a binary relation on A. We define the *irreflexive transitive closure* F^+ as

$$\begin{aligned} F_0 &\coloneqq F \\ F_{n+1} &\coloneqq F \cup \{(x, y) \mid \exists z \ xF_nzF_ny\} \\ F^+ &= \bigcup_{n \in \mathbb{N}} F_n. \end{aligned}$$

We then define the *(reflexive) transitive closure* F^* as

$$F^* \coloneqq F^+ \cup \mathrm{Id} \quad \text{where} \quad \mathrm{Id} \coloneqq \{(x, x) \mid x \in A\}.$$

This paper uses *partial orders* extensively, requiring the following notions and theorems.

Definition 2.2 (Partial order)**.** A pair $(A, \leq)$ with a set A and a relation $\leq \subseteq A \times A$ is called a partial order if for all $a, b, c \in A$:

- (Reflexivity) $a \leq a$;
- (Antisymmetry) $a \leq b \wedge b \leq a \implies a = b$;
- (Transitivity) $a \leq b \wedge b \leq c \implies a \leq c$.

Definition 2.3 (Maximal element)**.** Let $(A, \leq)$ be a partial order. We say that an element $a \in A$ is *maximal* in A if

$$\forall x \in A \ a \leq x \implies a = x.$$

Given a finite subset $X \subseteq A$, we let $\max(X)$ denote the set of maximal elements of $(X, \leq)$.

Proposition 2.1. *Let U and V be finite downwards closed subsets of the partial order $(A, \leq)$. Then we have that*

$$U = V \quad \textit{iff} \quad \max(U) = \max(V).$$

Proof. Trivially, we have that if $U = V$ then $\max(U) = \max(V)$.

For the other direction we assume that $\max(U) = \max(V)$. Without loss of generality, assume $x \in U$. We will prove that $x \in V$. There are 2 cases, either $x \in \max(U)$ or $x \notin \max(U)$. If $x \in \max(U)$ then $x \in \max(V)$ and $x \in V$. Now, if $x \notin \max(U)$ then there is a $y \in \max(U)$ such that $x \leq y$ by finiteness of U. Therefore $y \in V$ and $x \in V$ since V is downwards closed. Therefore, we conclude that

$$\max(U) = \max(V) \implies U = V. \qquad \square$$

Theorem 2.2 (Zorn's Lemma)**.** *Let $(A, \leq)$ be a partial order. Assume that every totally ordered subset of A has an upper bound. Then the set A contains at least one maximal element.*

The definition of Petri nets canonically uses a *multiset* (also known as a *bag*) to represent a set that can contain an element multiple times.

Definition 2.4 (Multiset)**.** Call a multiset a mapping $m : A \to \mathbb{N}$ for some set A. We define the elementary binary operations with m and n multisets on the set A:

$$\begin{aligned} m + n &\coloneqq \{(a, m(a) + n(a)) \mid a \in A\}, \\ m \setminus n &\coloneqq \{(a, \max(m(a) - n(a), 0)) \mid a \in A\}. \end{aligned}$$

Since multisets behave much like sets, we additionally define the standard set operations such that multisets behave like regular sets if $\forall a \in A \ m(a) \leq 1 \wedge n(a) \leq 1$:

$$\begin{aligned} m \cup n &\coloneqq m + (n \setminus m), \\ m \cap n &\coloneqq m \setminus (n \setminus m). \end{aligned}$$

The membership, subset and proper subset relations are then defined as:

$$\begin{aligned} a \in m &\iff m(a) > 0, \\ m \subseteq n &\iff m \cap n = m, \\ m \subset n &\iff m \subseteq n \wedge n \neq m. \end{aligned}$$

To be able to apply the operations above with a multiset and a normal set X as operands, we map X to a multiset given by

$$\{(a, 1) \mid a \in X\} \cup \{(a, 0) \mid a \in A \setminus X\}.$$

Additionally, we represent multisets as sets possibly containing the identical elements multiple times.

The definitions above are extended to the case where a multiset m can contain an element infinitely many times, in which case $m : A \to \mathbb{N}_\infty$, where $\mathbb{N}_\infty$ is the set of natural numbers including infinity.

Sometimes, we will look at the graph structures of Petri nets. Therefore, we introduce the following notions.

Definition 2.5 (Directed graph)**.** A directed graph is a pair $(\mathcal{N}, \mathcal{E})$ where $\mathcal{N}$ is the set of *nodes* and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of *edges*.

Definition 2.6 (Walk, path & cycle)**.** A sequence $(x_1, \ldots, x_n)$ of nodes in a directed graph $G = (\mathcal{N}, \mathcal{E})$ is called a *walk* if $\forall 0 < i < n \ (x_i, x_{i+1}) \in \mathcal{E}$. A walk is called a *path* if all nodes in the walk are distinct. A walk is called a *cycle* if $(x_1, \ldots, x_{n-1})$ is a path and $x_1 = x_n$.

Definition 2.7 (Strongly connected component)**.** Given a directed graph $G = (\mathcal{N}, \mathcal{E})$, we call a *strongly connected component* a maximal set of nodes $\{x_i\} \subseteq \mathcal{N}$ such that there exists a path between any ordered pair of nodes (x_i, x_j) with $i \neq j$.

2.2 Probability

The definitions of probability in this paper are based on the canonical measure-theory approach, following the example set by [3].

Definition 2.8 (Measurable space & σ-algebra)**.** Let U be a non-empty set. A set $\mathfrak{F}$ of subsets of U is called a *σ-algebra* if $\emptyset \in \mathfrak{F}$ and $\mathfrak{F}$ is closed under complement and countable union. The pair $(U, \mathfrak{F})$ is then called a *measurable space*. The elements of $\mathfrak{F}$ are called *measurable subsets* of $(U, \mathfrak{F})$.

Definition 2.9 (Measurable mapping)**.** If $(U, \mathfrak{F})$ and $(V, \mathfrak{G})$ are measurable spaces, then a mapping $\phi : U \to V$ is called a *measurable mapping* if $\phi^{-1}(A) \in \mathfrak{F}$ for every $A \in \mathfrak{G}$. Adopting the terminology from probability theory, measurable mappings are also called *random variables*.

Definition 2.10 (Induced σ-algebra)**.** Let $(U, \mathfrak{F})$ be a measurable space. For any measurable subset A of U there is a measurable space $\left(A, \mathfrak{F}^A\right)$, where we say that $\mathfrak{F}^A$ is *induced* by $\mathfrak{F}$ and is defined by

$$\mathfrak{F}^A \coloneqq \{B \in \mathfrak{F} \mid B \subseteq A\}.$$

Definition 2.11 (Probability space & probability measure)**.** If $(U, \mathfrak{F})$ is a measurable space, then the triple $(U, \mathfrak{F}, \mathbb{P})$ is said to be a *probability space* if $\mathbb{P}$ is a non-negative set function $\mathbb{P} : \mathfrak{F} \to \mathbb{R}$ such that $\mathbb{P}(\emptyset) = 0$ and $\mathbb{P}(U) = 1$ and for any sequence $(A_n)_{n\geq 0}$ of pairwise disjoint measurable subsets, we have

$$\mathbb{P}\left(\bigcup_{n\geq 0} A_n\right) = \sum_{n\geq 0} \mathbb{P}(A_n).$$

The map $\mathbb{P}$ is then called a *probability measure* or simply a *probability* and can be determined by how it acts on the singletons $\{x\}$ with $x \in U$. Therefore, we write $\mathbb{P}(x) = \mathbb{P}(\{x\})$, yielding $\sum_{x\in U} \mathbb{P}(x) = 1$.

Chapter 3

Concurrency

In this chapter we will present the classes of models for concurrent systems. Additionally, we treat the application of concurrency in the field of physics.

3.1 Models of Concurrency

There are multiple models of concurrency, each highlighting different characteristics of concurrent systems. Most of these models take as their atomic element some indivisible (sequential) change (usually called events, actions, transitions or symbols) [10]. These changes are then composed into more complex concurrent structures by specifying an order of causality between them. This order of causality in a concurrent system is fundamentally a partial order [7]. If this partial order is used, we say that *true-concurrent* or *non-interleaving* semantics are used. Alternatively, the events can be totally ordered. In this case we call the semantics *interleaving*.

Sassone, Nielsen, and Winskel [10] have introduced three axes along which to compare models of concurrency. The first axis distinguishes *system* and *behaviour* models. System models have a representation of the state of a system, while behaviour models are not concerned with state itself, but only with the patterns of states. The second axis describes whether the system uses true-concurrent or interleaving semantics. Finally, the third axis distinguishes *branching time* models from *linear time* models. Branching models have a representation of the choices in the system, while linear models do not.

In this paper, Petri nets and event structures will be used as models for concurrency. In the taxonomy from [10], Petri nets are an example of a system non-interleaving branching time model [10]. Similarly, event structures are a behaviour non-interleaving branching time model [10]. Although both can also be equipped with interleaving semantics.

3.2 Concurrency in Physics

We end the chapter by briefly considering some connections between concurrency theory and physics.

The necessity of true-concurrency does not only arise for practical synchronisation reasons, but true-concurrency is in fact a fundamental property of the universe that follows from the theory of special relativity. The theory of special relativity is based on two main postulates [6]:

1. the laws of physics are of the same form in all inertial reference frames[1];
2. the speed of light in vacuum has the same value c in all inertial reference frames.

The constant speed of light implies that vector addition cannot be used to add velocities. Instead, Einstein [6] proposed to use Lorentz transformations. The remarkable property of Lorentz transformations with respect to concurrency is that it introduces *time dilation*, effectively stretching time depending on the velocity of the observer.

Consider two events separated by space a and b, which are simultaneous in one observer's frame. If a second observer is moving towards a, the Lorentz transformation results in a happening before b from the perspective of this observer, in which case they are no longer simultaneous.[2] It follows from the first postulate that there exists no absolute reference frame that can be regarded as the sole truth. Therefore, these events are fundamentally true-concurrent.

[1] Inertial means non-accelerating.
[2] It must be noted that causality cannot be affected by Lorentz transformations.

Chapter 4

Petri Nets

In this section, the basic theory of Petri nets will be introduced. Then, the concepts of confusion and of persistent places will be discussed. These notions will have a crucial role in Chapters 7 and 8. Unless otherwise specified, this chapter follows the definitions and notation from [5].

4.1 Preliminaries on Petri Nets

Petri nets were first introduced by C. A. Petri in 1962 [9] to model concurrent processes such as chemical processes. Since then, Petri nets have been generalised to describe any kind of concurrent process.

Definition 4.1 (Petri net)**.** A Petri net is a tuple (P, T, F) where

- P is the set of places,
- T is the set of transitions, which is disjoint from P,
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation.

A *marked Petri net* is a tuple (P, T, F, m), where (P, T, F) is a Petri net and m is a multiset $m \in \mathbb{N}^P$, that is, m is a mapping from P to $\mathbb{N}$. If $p \in m$, then p is said to be *marked* at m.

Adopting terminology from graph theory, the set $P \cup T$ is called the set of *nodes*. If x is a node of N then we write $x \in N$. Given a node $x \in N$, we define the *preset* ${}^\bullet x$ and *postset* $x^\bullet$ of x as

$$ {}^\bullet x \coloneqq \{y \mid (y, x) \in F\} \quad \text{and} \quad x^\bullet \coloneqq \{z \mid (x, z) \in F\}. $$

Additionally, we define the *initial places* ${}^\circ N$ and the *final places* N° of the Petri net N as

$$ {}^\circ N \coloneqq \{p \in P \mid {}^\bullet p = \emptyset\} \quad \text{and} \quad N^\circ \coloneqq \{p \in P \mid p^\bullet = \emptyset\}. $$

A Petri net $N' = (P', T', F')$ is called a subnet of a Petri net $N = (P, T, F)$, written $N' \subseteq N$ if $P' \subseteq P \wedge T' \subseteq T \wedge F' \subseteq F$.

Furthermore, we define the *causality relation* as the transitive closure of the flow relation:

$$\preceq := F^*.$$

To remove nodes from a Petri net, we define the set difference on a Petri net $N = (P, T, F)$ as

$$N \setminus X = (P', T', F'),$$

with

$$P' = P \setminus X, \quad T' = T \setminus X, \quad F' = F \cap ((P' \times T') \cup (T' \times P')).$$

All operations on Petri nets in this paper are implicitly extended to marked Petri nets, restricting m to the elements of P if marked places are removed.

In modelling a concurrent process with a Petri net, one associates a place with each condition that might hold during the execution process. The state at a given time in the process is given by the set of places in the marking. The transitions then represent the ability of the system to change its state from one marking to another.

Definition 4.2 (Enabling & firing)**.** A transition t is *enabled* at the marking m, written $m \xrightarrow{t}$ if ${}^\bullet t \subseteq m$. If a transition is *fired*, written $m \xrightarrow{t} m'$, then $m' = (m \setminus {}^\bullet t) + t^\bullet$. A finite sequence of firings is called a *firing sequence*. A firing sequence $m_0 \xrightarrow{t_1} \dots \xrightarrow{t_n} m_n$ is abbreviated $m_0 \xrightarrow{t_1 \cdots t_n} m_n$ or $m \to^* m'$. Given an initial marking, firing sequences can be uniquely identified by a sequence of transitions.

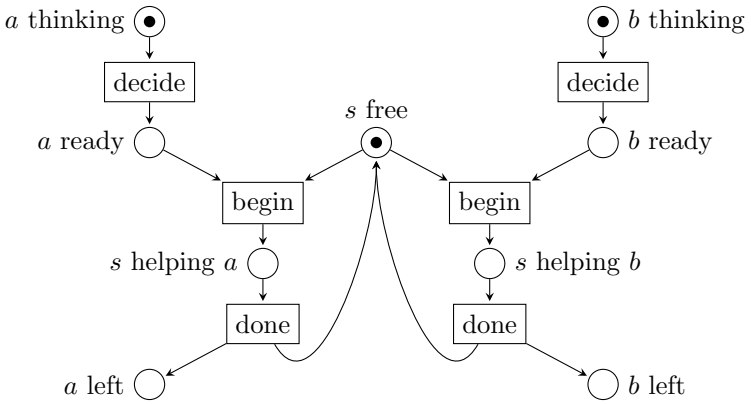
As a result of these definitions, places can hold any (natural) number of tokens. A marked Petri net is said to be 1-*safe* if it is impossible to reach a marking via a firing sequence where a single place holds two or more tokens. In this paper, only 1-safe Petri nets are considered.

Let us illustrate the use of Petri nets with a simple example. Consider a shop with a single shopkeeper s and two customers, labelled a and b. Before they can buy something, the customers have to decide what to buy. When they are ready to order, they complete the transaction with the shopkeeper, after which the customers leave the shop.

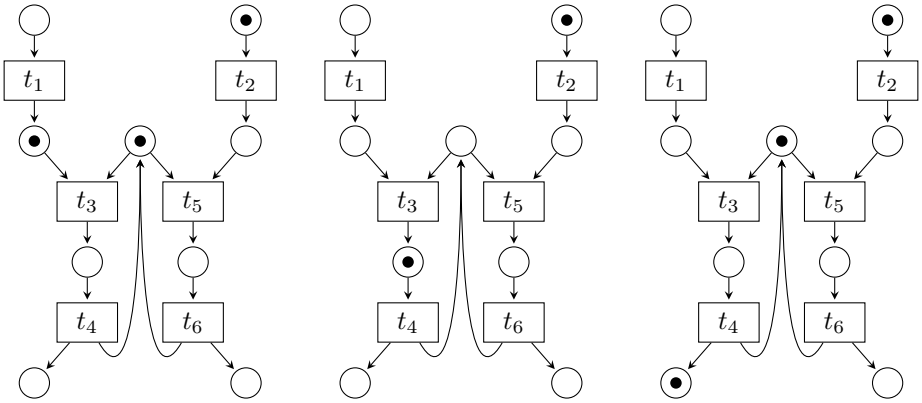
In this model, each customer has 3 possible states: "`thinking`" (about what to buy), "`ready`" (to order) and "`left`" (the shop). The shopkeeper can be either "`free`" or "`busy`" helping a customer. Initially, each customer is in the "`thinking`" state and the shopkeeper is in the "`free`" state. When the customer decide what they want, they transition from "`thinking`" to "`ready`". When the shopkeeper and a customer are "`free`" and "`ready`", respectively, they transition to "`busy`". After an arbitrary amount of time, the transaction is completed and the customer's state is set to "`done`" and the shopkeeper returns to being "`free`".

The marked Petri net corresponding to this process is shown in Figure 4.1. In particular, Figure 4.1a shows the initial state of the process. Figures 4.1b

to 4.1d then show the evolution of the markings when customer a makes a decision, is helped by the shopkeeper s, and leaves the shop. Places are represented by circles, transitions are represented with rectangles and the flow relation is represented by edges between the nodes. Places contain a number of dots corresponding to their marking.



(a) The net with the initial marking and descriptions for the places and transitions.



(b) The net after firing t_1.

(c) The net after firing t_1 and t_3.

(d) The net after firing t_1, t_3 and t_4.

Figure 4.1: The Petri net corresponding to the shop example. The net starts out with the initial marking in Figure 4.1a. Figures 4.1b to 4.1d show the dynamics of the system, after the transitions t_1, t_3 and t_4 are fired, respectively.

As defined above, a transition can only be fired when all the places in its preset are marked. When a transition is fired, the places in the preset are removed from the marking and the places in the postset are added. If transitions share one or more places in their preset, a choice between the transitions has to made. As we will see, choices will play a central role in this paper.

Definition 4.3 (Conflict)**.** Two transitions, t_1 and t_2, are in *direct conflict*, written $t_1 \#_0 t_2$, if

$$t_1 \neq t_2 \wedge {}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset.$$

Additionally, two nodes $x, y \in N$ are in *conflict*, denoted by $x \# y$, if any of their "ancestors" are conflict. Formally, $\#$ is defined by

$$\forall x, y \in N \quad x \# y \text{ iff } \exists e \preceq x \ \exists e' \preceq y \ e \#_0 e'.$$

It is possible that a node x has nodes in its preset which are in conflict with each other. By Definition 4.3, this implies $x \# x$. In this case, we say that x is in conflict with itself and that there is *auto-conflict* in the net.

We define the previously discussed concept of confusion in the context of Petri nets.

Definition 4.4 (Confusion)**.** A 1-safe marked net has confusion if there exists a reachable marking m and transitions t, u, v such that the symmetric or asymmetric case holds.

1. In the symmetric case,
   (a) t, u, v are enabled at m,
   (b) ${}^\bullet t \cap {}^\bullet u \neq \emptyset \neq {}^\bullet u \cap {}^\bullet v$ and
   (c) ${}^\bullet t \cap {}^\bullet v = \emptyset$.
2. In the asymmetric case,
   (a) t and v are enabled at m,
   (b) u is not enabled at m but becomes enabled after the firing of t,
   (c) ${}^\bullet t \cap {}^\bullet v = \emptyset$ and ${}^\bullet u \cap {}^\bullet v \neq \emptyset$.

When a net does not have confusion, it is said to be *confusion-free*.

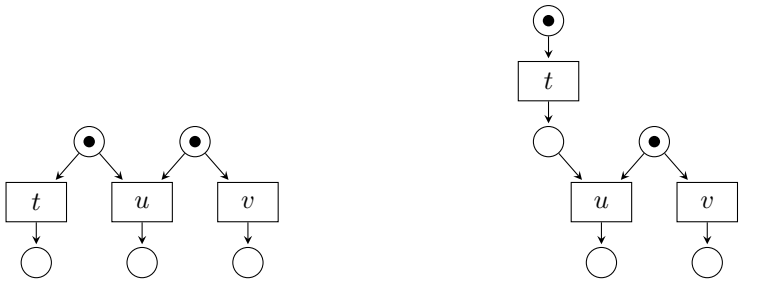
Consider the net in Figure 4.2a. If t is fired, u is no longer enabled, removing the choice between u and v. This net has symmetric confusion. Similarly, if t is fired in the net in Figure 4.2b, u becomes enabled and a choice between u and v is added. This is an example of asymmetric confusion.

4.2 Occurrence Nets

An occurrence net is an acyclic Petri net without auto-conflict. They will be instrumental in connecting Petri nets to event structures in Chapter 5.

Definition 4.5 (Occurrence net)**.** A Petri net $N = (P, T, F)$ is called an *occurrence net* if it satisfies the following properties:

- $(P \cup T, \preceq)$ is a partial order;
- for every $x \in P \cup T$, the set $\{y \in P \cup T \mid y \preceq x\}$ is finite;



(a) Symmetric confusion. (b) Asymmetric confusion.

Figure 4.2: The two simple cases of confusion.

- for every $p \in P$, $|{}^\bullet p| \leq 1$; and
- $\#$ is irreflexive, meaning that there is no auto-conflict.

The definition above is illustrated in Figures 4.3 and 4.4. Figure 4.3 shows nets that are not occurrence nets, since they are not acyclic, have places with a preset of size larger than 1 or contain nodes which are in conflict with themselves. In contrast, Figure 4.4 shows examples of nets that are occurrence nets.

To identify the interleaving traces of Petri nets, we introduce equivalence classes called *processes*.

Definition 4.6 (Deterministic occurrence net)**.** An occurrence net is called *deterministic* if there is no conflict.

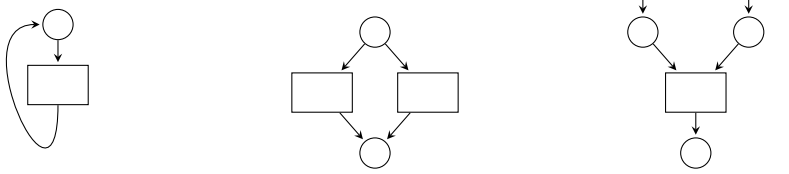
Definition 4.7 (Process)**.** A *deterministic nonsequential process* or process is a mapping $\pi : \mathcal{D} \to N$ from a deterministic occurrence net $\mathcal{D}$ to a Petri net N that preserves preset and postset and such that $\pi({}^\circ\mathcal{D})$ is the initial marking of N. Firing sequences in N are equivalent if they are images of maximal firing sequences in the same deterministic occurrence net $\mathcal{D}$. A process is called *maximal* if its maximal firing sequences are maximal in N.

For example in the net in Figure 4.1, the firing sequences

$$(t_2, t_1, t_3, t_4),$$
$$(t_1, t_2, t_3, t_4),$$
$$(t_1, t_3, t_2, t_4),$$
$$(t_1, t_3, t_4, t_2),$$

are all considered to be equivalent and belong to the equivalence class given by the deterministic occurrence net $U = (P', T', F')$ with

$$T' = \{t_1, t_2, t_3, t_4\},$$
$$P' = \bigcup_{t \in T'} ({}^\bullet t \cup t^\bullet),$$
$$F' = F \cap ((P' \times T') \cup (P' \times T')).$$



(a) A net with a cycle.

(b) A net with a place such that $|{}^\bullet p| > 1$.

(c) A net with auto-conflict.

Figure 4.3: Nets that are *not* occurrence nets since they each do not satisfy one of the necessary properties. Adapted from [1].

(a) A tree-like net is an occurrence net.

(b) Concurrent processes are possible as long as conflicting processes are not joined.

Figure 4.4: Examples of occurrence nets. Adapted from [1].

4.3 Unfolding

A 1-safe marked Petri net can be *unfolded* into an occurrence net while preserving the possible firing sequences. This occurrence net can be used as a canonical representation of the dynamics of the net; when two 1-safe nets have the same behaviour, their occurrence nets will be equal [13, Thm. 4.7]. Unfolding is also the first step in converting a Petri net into an event structure, as will be discussed in Section 5.2. The construction of the unfolding detailed below is based on [8].

The unfolding is constructed using an equivalence relation $\equiv$ on firing sequences. To define $\equiv$, we take any firing sequence

$$\sigma = m_0 \xrightarrow{t_0 \cdots t_n} m_{n+1},$$

where m_0 is the initial marking. The equivalence relation $\equiv$ will be defined from two auxiliary relations. First, we say that $\sigma \equiv_{(1)} \sigma'$ if there is $i \leq n-1$ such that

$$\sigma' = m_0 \xrightarrow{t_0 \cdots t_{i-2}} m_{i-1} \xrightarrow{t_i} m_i' \xrightarrow{t_{i-1}} m_{i+1} \xrightarrow{t_{i+1} \cdots t_n} m_{n+1},$$

that is, the positions of t_i and t_{i-1} are swapped, changing the marking m_i to some other marking m_i'. Second, we say that $\sigma \equiv_{(2)} \sigma''$ if

$$\sigma'' = m_0 \xrightarrow{t_0 \cdots t_{n-2}} m_{n-1} \xrightarrow{t_n} m_{n+1}',$$

where t_{n-1} is removed from the firing sequence, changing the marking m_{n+1} to some other marking m_{n+1}'. We then define $\equiv$ as the reflexive symmetrical transitive closure of $\equiv_{(1)}$ and $\equiv_{(2)}$:

$$\equiv \coloneqq \left(\equiv_{(1)} \cup \equiv_{(2)} \cup \equiv_{(2)}^{-1} \right)^*.$$

We denote the set of equivalence classes of $\equiv$ with S. Note that all firing sequences in an equivalence class $s \in S$ of $\equiv$ have the same final transition which we identify as t_s. Additionally, given firing sequences

$$\begin{aligned} \sigma &= m_0 \xrightarrow{t_0 \cdots t_{n-1}} m_n \xrightarrow{t_n \cdots t_{k-1}} m_k \xrightarrow{t_k \cdots t_{l-1}} m_l, \\ \sigma' &= m_n \xrightarrow{t_n \cdots t_{k-1}} m_k, \end{aligned}$$

abusing notation, we freely write σ as

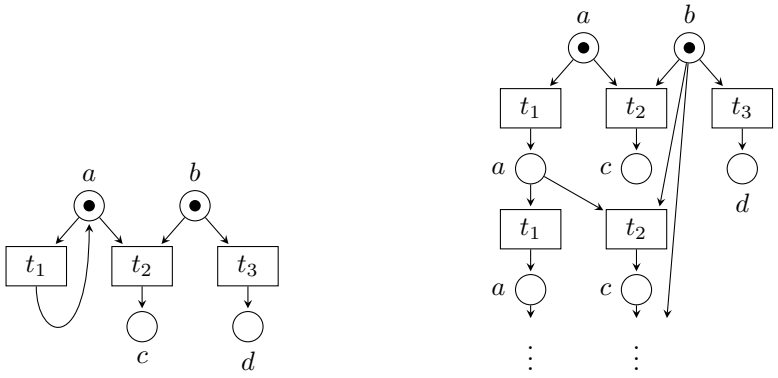
$$\sigma = m_0 \xrightarrow{t_0 \cdots t_{n-1}} \sigma' \xrightarrow{t_k \cdots t_{l-1}} m_l.$$

The unfolding of a 1-safe Petri net (P, T, F) is then an occurrence net $U =$

(P', T', F'), where

$$
\begin{aligned}
T' &= S, \\
P' &= \{(e, p) \mid s \in S \wedge p \in t_s^\bullet\} \cup \{(\emptyset, p) \mid p \in m\}, \\
F' &= \{(s, (s, p')) \mid (s, p') \in P'\} \\
&\quad \cup \left\{(p', s) \;\middle|\; p' = (s', p) \wedge s = s' \xrightarrow{t_n} m_{n+1} \wedge p \in {}^\bullet t_s\right\} \\
&\quad \cup \left\{(p', s) \;\middle|\; p' = (\emptyset, p) \wedge s = m_0 \xrightarrow{t_0} m_1 \wedge p \in {}^\bullet t_0\right\}.
\end{aligned}
$$

A simple Petri net and its unfolding are illustrated in Figure 4.5, Only a prefix of the unfolding is shown since t_1 can be fired any number of times in the original net.



(a) A net containing a loop.

(b) The prefix of the unfolding of the net.

Figure 4.5: A cyclic net and its unfolding. Adapted from [1].

Chapter 5

Event Structures

Event structures are used as an intermediate structure in the process of attaching probabilities to Petri nets in the methods by Varacca, Völzer, and Winskel [12] and Abbes and Benveniste [2]. This chapter discusses event structures, their connection to Petri nets and the concept of local finiteness.

5.1 Preliminaries on Event Structures

Definition 5.1 (Event Structure)**.** An event structure is a triple

$$\mathcal{E} = (E, \preccurlyeq, \#)$$

where E is the set of *events*, $\preccurlyeq$ is the *causality* relation and $\#$ is the *conflict* relation. An event structure $\mathcal{E}$ must satisfy the following properties:

- the set E is at most countable;
- the tuple $(E, \preccurlyeq)$ is a partial order such that for any event e the set $\{e' \in E \mid e' \preccurlyeq e\}$ is finite and
- the relation $\#$ is symmetric and irreflexive, and satisfies

$$\forall x, y, z \in E \ (x \# y \wedge y \preccurlyeq z) \implies x \# z.$$

Definition 5.2 (Prefix & Configuration)**.** A subset $A \subseteq E$ is called a *prefix* if it is downwards closed, that is, if

$$\forall x \in E \ \forall y \in A \quad x \preccurlyeq y \implies x \in A.$$

Additionally, a prefix v is called a *configuration* if it is *conflict-free*:

$$\# \cap (v \times v) = \emptyset.$$

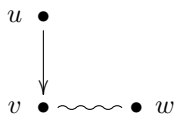


Figure 5.1: An example of an event structure. It corresponds to the asymmetric confusion Petri net in Figure 4.2b.

A configuration v represents a valid execution of the system, since each event in v only depends on events in v and no two events in conflict can be in the same configuration. Therefore, there is a valid order in which the events in v can be executed.

We say that two configurations x, y are *compatible* if $x \cup y$ is a configuration. We denote the set of finite configurations with $\mathcal{V}_\mathcal{E}$. The elements of $\mathcal{V}_\mathcal{E}$ can be partially ordered based on the events they include. Additionally, we define the set of maximal configurations $\Omega_\mathcal{E}$. By applying Zorn's lemma on the partial order $(\mathcal{V}_\mathcal{E}, \subseteq)$, we conclude that this set is non-empty [1].

A subset of events $F \subseteq E$ induces a sub-event structure $(F, \preccurlyeq_F, \#_F)$ with

$$\preccurlyeq_F = \preccurlyeq \cap (F \times F), \quad \#_F = \# \cap (F \times F).$$

To denote the finite and maximal configurations of this event structure, abusing notation we will freely write $\mathcal{V}_F$ and Ω_F, respectively, without explicitly stating the event structure.

Given an event structure $\mathcal{E} = (E, \preccurlyeq, \#)$, The smallest configuration containing the element $e \in E$ is denoted

$$[e] \coloneqq \{e' \in E \mid e' \preccurlyeq e\}.$$

Building on this definition, the smallest configuration enabling e is denoted

$$[e[\coloneqq [e] \setminus \{e\}.$$

Definition 5.3 (Immediate conflict)**.** The *immediate conflict* relation $\#_\mu$ on E is defined as

$$\forall e, e' \in E \qquad e \#_\mu e' \text{ iff } ([e] \times [e']) \cap \# = \{(e, e')\}.$$

Immediate conflict is analogous to direct conflict in Petri nets, as it describes where choices must be made in the execution of the event structure.

An example of the graphical representation of event structures is given in Figure 5.1. Events are represented by a dot ($\bullet$). Causality is given by the transitive reflexive closure of the arrows ($\longrightarrow$) and immediate conflict is denoted by wavy lines ($\rightsquigarrow$).

Definition 5.4 (Future of a configuration)**.** Let $\mathcal{E} = (E, \preccurlyeq, \#)$ be an event structure. Call the *future* of a configuration v the event structure $\mathcal{E}^v$ induced by

$$E^v \coloneqq \{e \in E \setminus v \mid \forall e' \in v \ \neg(e \# e')\},$$

which is the set of events not in conflict with any of the events in the configuration and not in the configuration itself.

5.2 From Petri Nets to Event Structures

Petri nets and event structures are both models for concurrent processes. However, event structures lack a representation of the state of a system, since the only atomic elements are events, which are analogous to transitions. Nevertheless, Petri nets can be mapped to event structures by using occurrence nets as an intermediate step.

Let $N = (P, T, F)$ be a 1-safe Petri net. As shown by Winskel [13, Thm. 4.7], N has an unfolding (U, ρ), where $U = (P', T', F')$ is an occurrence net. Nielsen, Plotkin, and Winskel [8] have defined the map that converts an occurrence net to an event structure:

$$\xi(U) \coloneqq (T',\ F^* \cap (T' \times T'),\ \#_U \cap (T' \times T')).$$

Figure 5.2a shows the unfolding of the Petri net in Figure 4.1. The event structure obtained by applying ξ to the unfolding is shown in Figure 5.2b.

5.3 Confusion in Event Structures

The notion of confusion can also be translated from the theory of Petri nets to event structures. This will be useful for some of the definitions later on.

In the symmetric case of confusion, there exist three transitions t, u and v which we will directly map to three events t, u and v. Since ${}^\bullet t \cap {}^\bullet u \neq \emptyset \neq {}^\bullet u \cap v$ in the Petri net model, we conclude that $t \#_\mu u$ and $u \#_\mu v$. Furthermore, we see that $\neg(t \#_\mu v)$, since ${}^\bullet t \cap {}^\bullet v = \emptyset$. The condition for symmetric confusion is then

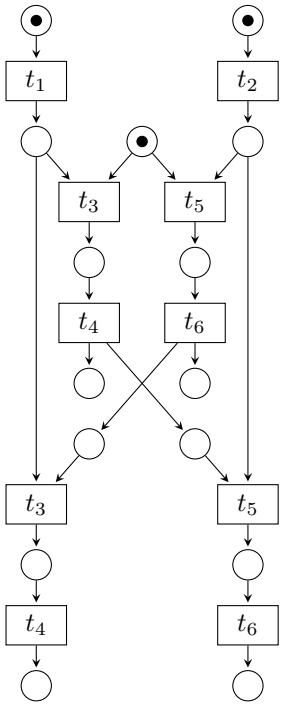
$$\exists t, u, v \in E \quad t \#_\mu u \wedge u \#_\mu v \wedge \neg(t \#_\mu v).$$

In other words, the immediate conflict relation must be transitive for the Petri net to be confusion-free.

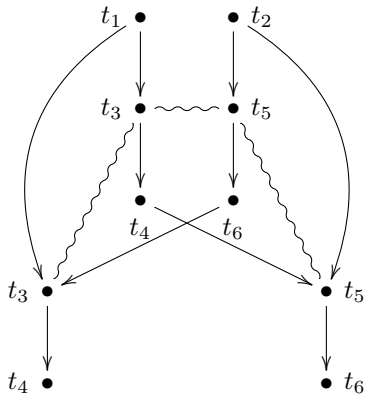
Using similar reasoning as before, we can see that there are two events u and v such that $u \#_\mu v$ in the asymmetric case of confusion. Furthermore, there is some event t that is maximal in $[u[$ and $t \notin [v[$. By Proposition 2.1, the configurations $[u[$ and $[v[$ are equal if and only if $\max([u[) = \max([v[)$. Therefore, the condition for asymmetric confusion is

$$\exists u, v \in E \quad u \#_\mu v \wedge [u[\neq [v[.$$

The corresponding condition for an event structure to be confusion-free is then that $\forall u, v \in E \quad u \#_\mu v \implies [u[= [v[$.



(a) The occurrence net corresponding to the Petri net in Figure 4.1.



(b) The event structure corresponding to the occurrence net in Figure 5.2a.

Figure 5.2: The occurrence net and event structure corresponding to the Petri net in Figure 4.1.

(a) Symmetric confusion (b) Asymmetric confusion

Figure 5.3: The minimal confused event structures.

Theorem 5.1 (Confusion)**.** *An event structure $\mathcal{E} = (E, \preccurlyeq, \#)$ has confusion if at least one of the following conditions holds:*

1. *(Symmetric)* $\exists t, u, v \in E \;\; t \#_\mu u \wedge u \#_\mu v \wedge \neg(t \#_\mu v)$*;*
2. *(Asymmetric)* $\exists u, v \in E \;\; u \#_\mu v \wedge [u[\neq [v[$*.*

Examples of confused event structures are shown in Figure 5.3, which are direct translations from the Petri nets in Figure 4.2.

5.4 Locally Finite Event Structures

The concept of locally finite event structures was introduced by Abbes and Benveniste [2] to describe the class of event structures, and by extension Petri nets, that their method applies to. Intuitively, an event structure is locally finite if for every event e there is a finite number of events that directly or indirectly influence whether e can be fired.

Definition 5.5 (Stopping prefix)**.** A prefix B of $\mathcal{E}$ is called *stopping* if it is closed under immediate conflict.

Definition 5.6 (Locally finite)**.** An event structure $\mathcal{E} = (E, \preccurlyeq, \#)$ is called *locally finite* if for each event $e \in \mathcal{E}$, there exists a finite stopping prefix containing e.

A Petri net is called locally finite if the event structure $\xi(U)$ corresponding to its unfolding U is locally finite.

The difference between a locally finite and a non-locally finite event structure is illustrated in Figure 5.4. In Figure 5.4b, the right-most event is in immediate conflict with infinitely many events in the middle column. Any finite prefix containing that event is therefore not stopping since it would not be closed under immediate conflict. In Figure 5.4a, every event is only in immediate conflict with two other events.

(a) A locally finite event structure. (b) A non-locally finite event structure.

Figure 5.4: Two similar event structures: one locally finite and one non-locally finite.

Chapter 6

Randomizing Confusion-Free Event Structures

This chapter discusses a method of attaching local probabilities to confusion-free event structures introduced by Varacca, Völzer, and Winskel [12]. Then, we attempt to apply the same method to confused event structures to understand why the method fails. The limitations of this method motivate the branching cells method described in Chapter 7.

6.1 Probabilistic Event Structures

With the definitions of event structures in place, *probabilistic event structures with independence* can be defined. This is done in the way described in [12]. The idea here is to define probability to local parts of the event structure that generate a sensible probability distribution for the configurations. We will first only consider confusion-free event structures.

To simplify some upcoming definitions, the concept of a *covering* is introduced.

Definition 6.1 (Covering)**.** Let $x, x' \in \mathcal{V}_\mathcal{E}$ be two configurations. We say that x *covers* x' if there exists an event $e \notin x'$ such that $x = x' \cup \{e\}$. A covering at x is a maximal non-empty set of pairwise incompatible configurations that cover x.

The previously mentioned "sensible" probability for the configurations must satisfy three conditions: normality, conservation and independence. Such a probability distribution is called a *configuration valuation with independence*.

Definition 6.2 (Configuration valuation with independence)**.** A mapping $v : \mathcal{V}_\mathcal{E} \to [0, 1]$ is called a configuration valuation with independence if it satisfies

1. (Normality) $v(\emptyset) = 1$;
2. (Conservation) if C is a covering at x, then $\sum_{c \in C} v(c) = v(x)$;
3. (Independence) if x and y are compatible then $v(x) \cdot v(y) = v(x \cup y) \cdot v(x \cap y)$.

The first condition simply states that the empty configuration must always occur, since it is the starting point of every configuration. The second condition implies that the sum of the probabilities of the maximal configurations is 1. Finally, the condition of independence states that the valuations for compatible configurations must be probabilistically independent.

To create a *probabilistic event structure with independence*, we attach a configuration valuation with independence to an event structure.

Definition 6.3 (Probabilistic event structure with independence)**.** A probabilistic event structure with independence is a pair $(\mathcal{E}, v)$ with a confusion-free event structure $\mathcal{E}$ and a configuration valuation v.

Now we define local probability distributions to match the global probability distribution defined above. To this end, *cells* are defined to represent the sets of events where choices have to be made. The local probability distributions of the event structure will be attached to each of these cells.

Definition 6.4 (Cell)**.** A cell is a maximal non-empty set c of events such that $e, e' \in c$ implies $e \#_\mu e'$ and $[e[= [e'[$.

Varacca, Völzer, and Winskel [12, Definition 2.3] give a different yet equivalent definition of confusion-freeness based on cells. Here, this definition is treated as a theorem following from Theorem 5.1.

Theorem 6.1 (Confusion-free)**.** *An event structure is confusion-free if and only if its cells are closed under immediate conflict.*

Proof. Let $\mathcal{E} = (E, \leq, \#)$ be an event structure. We first prove that the cells of an event structure with confusion are not closed under immediate conflict. Consider the symmetric case of confusion, such that

$$\exists t, u, v \in E \quad t \#_\mu u \wedge u \#_\mu v \wedge \neg(t \#_\mu v). \tag{6.1}$$

Therefore, a cell c containing t cannot contain v since $\neg(t \#_\mu v)$. Hence, c is not closed under $\#_\mu$.

In the asymmetric case of confusion, we have

$$\exists u, v \in E \quad u \#_\mu v \wedge [u[\neq [v[. \tag{6.2}$$

Therefore, a cell c containing u cannot contain v, since $[u[\neq [v[$. Hence, c is not closed under $\#_\mu$.

Now we prove that the existence of a cell which is not closed under $\#_\mu$ implies that the net has confusions, we assume that there exists a cell c that is not closed under $\#_\mu$. Then, there must exist two events t and u such that

$t \#_\mu u$ with $t \in c$ and $u \notin c$. Therefore, one of two conditions must hold: $\exists x \in c\ \neg(x \#_\mu u)$ or $\exists x \in c\ [x[\neq [u[$. In the first case, Equation (6.1) holds, therefore the net is symmetrically confused.

Assume the first condition does not hold, so $\forall x \in c\ x \#_\mu u$. Now the second condition must hold, therefore we conclude $\exists x \in c\ x \#_\mu u \wedge [x[\neq [u[$. Therefore Equation (6.2) holds and the net is asymmetrically confused. □

A *cell valuation* then attaches a probability to each of the choices in the cell.

Definition 6.5 (Cell valuation)**.** A cell valuation on a confusion-free event structure $\mathcal{E} = (E, \preccurlyeq, \#)$ is a mapping $p : E \to [0, 1]$ such that for every cell c, we have $\sum_{x \in c} p(x) = 1$.

These local probabilities then give rise to a configuration valuation by multiplying the probabilities of the events in the configurations.

Proposition 6.2 (Varacca, Völzer, and Winskel [12, Prop. 2.8])**.** *Let p be a cell valuation and $v : \mathcal{V}_\mathcal{E} \to [0, 1]$ be mapping given by $v(x) = \prod_{e \in x} p(e)$. Then v satisfies normality, conservation and independence. Therefore, v is a configuration valuation with independence.*

Proposition 6.3 (Varacca, Völzer, and Winskel [12, Prop. 2.11])**.** *Let v be a configuration valuation with independence. Then there exists a cell valuation p such that $v(c) = \prod_{e \in x} p(e)$.*

As an example of a cell valuation, consider the net in Figure 6.1. To create a valid cell valuation for this event structure, the probabilities for the events in each cell must add up to 1. For example, we can attach the probabilities given in the table in Figure 6.1. To determine the configuration valuation we then multiply the probabilities of the events in the configuration. For example, the configuration $\{a, d\}$ has the valuation

$$v(\{a, d\}) = p(a) \cdot p(d) = \frac{3}{4} \cdot \frac{1}{2} = \frac{3}{8}.$$

The only covering for this configuration then consists of the configurations $\{a, d, e\}$ and $\{a, d, f\}$. The valuation of these configurations are

$$v(\{a, d, e\}) = \frac{1}{8} \quad \text{and} \quad v(\{a, d, f\}) = \frac{2}{8},$$

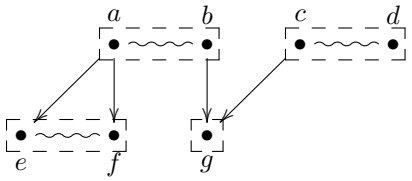
which indeed add up to $v(\{a, d\})$, as required by the condition of conservation. Furthermore, if we take the compatible configurations $v(\{a, f\})$ and $v(\{a, c\})$, we have that

$$v(\{a, f\}) \cdot v(\{a, c\}) = \frac{3}{4} \cdot \frac{2}{3} \cdot \frac{3}{4} \cdot \frac{1}{2} = \frac{3}{16}$$

and

$$\begin{aligned} v(\{a, c\} \cup \{a, f\}) \cdot v(\{a, c\} \cap \{a, f\}) &= v(\{a, c, f\}) \cdot v(\{a\}) \\ &= \frac{3}{4} \cdot \frac{2}{3} \cdot \frac{1}{2} \cdot \frac{3}{4} = \frac{3}{16}, \end{aligned}$$

as required by the condition of independence.



x	a	b	c	d	e	f	g
$p(x)$	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{2}{3}$	1

Figure 6.1: Confusion-free net with cells represented as dashed rectangles and an example of a cell valuation for this net.

6.2 Confusion in Probabilistic Event Structures

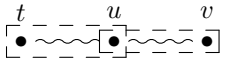
The approach discussed in the previous section works well for confusion-free events structures, but the case of confusion still has to be discussed.

Consider the nets in Figure 6.2. For the symmetric case of confusion, the cells overlap, meaning that the probabilities on the left and right event must be equal to each other. While possibly inconvenient, this does not directly pose a problem. Instead, the problem is that the condition of conservation is not satisfied. Assume that the events have the non-zero valuations $p(t)$, $p(u)$ and $p(v)$ and consider the configuration $\{t\}$ with the configuration valuation $v(\{t\}) = p(t)$. The only configuration in the covering at $\{t\}$ is $\{t, v\}$. This configuration has a valuation given by $v(\{t, v\}) = p(t) \cdot p(v)$. Therefore, we conclude that $v(\{t\}) > v(\{t, v\})$, meaning that the conservation condition is not satisfied.

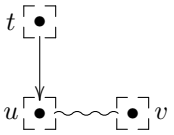
In the asymmetric case, we note that $p(t) = p(u) = p(v) = 1$, since all cells consist of a single event. Furthermore, the covering of the configuration $\{t\}$ is $\{\{t, u\}, \{t, v\}\}$. The sum of the valuations for the configurations in this covering is then

$$v(\{t, u\}) + v(\{t, v\}) = p(t) \cdot p(u) + p(t) \cdot p(v) = 2,$$

which violates the conservation condition since $v(\{t, u\}) + v(\{t, v\}) > v(\{t\})$.



(a) Symmetric confusion.



(b) Asymmetric confusion.

Figure 6.2: Cells of event structures with confusion

We conclude that the method by [12] does not work for nets with confusion because the cells are not closed under immediate conflict. To remedy this, a new type of cells needs to be defined that are closed under immediate conflict when the net has confusion. This approach is discussed in the next chapter.

Chapter 7

Branching Cells Method

In this chapter, we will discuss branching cells and how they are used to create probabilistic event structures with confusion. This approach was first introduced by Abbes [1], however, this chapter follows the definitions and notation from [2].

The method described in this chapter requires the following assumptions [2]:

1. For every event e, the configuration $[e]$ is finite.
2. For every $v \in \mathcal{V}_{\mathcal{E}}$, the set $\min_{\preccurlyeq}(E^v)$ contains finitely many events.
3. $\mathcal{E}$ is locally finite.

7.1 Branching Cells

The notion of a stopping prefix (see Definition 5.5) is used to define some preliminaries for branching cells.

Definition 7.1 (Stopped configuration)**.** A configuration v of $\mathcal{E}$ is said to be *stopped* if there is a stopping prefix B such that $v \in \Omega_B$.

A stopped configuration represents a configuration which resolves all conflict up to a certain point in the execution.

Definition 7.2 (Recursively stopped configuration)**.** A configuration v of $\mathcal{E}$ is said to be *recursively stopped* if there exists a finite non-decreasing sequence $(v_n)_{0 \leq n \leq N}$ of configurations, where $v_0 = \emptyset$, $v_N = v$ and for $n < N$, we have that $v_{n+1} \setminus v_n$ is a finite stopped configuration of the future $\mathcal{E}^{v_n}$ of v_n. The set of recursively stopped configurations of $\mathcal{E}$ is denoted $\mathcal{W}_{\mathcal{E}}$.

A recursively stopped configuration is similar to a stopped configuration, but at each step, it takes the future of the net into account. Consider for example the event structure in Figure 7.1. If a is fired, then the conflict between c and e is resolved because d is not in the future of the configuration $\{a\}$. The configuration $\{a, c\}$ is not stopped, since it is not maximal in the stopping prefix

$\{a, b, c, d, e\}$, however, it is recursively stopped since $\{a\}$ is stopped in the initial event structure and $\{c\}$ is stopped in $\mathcal{E}^{\{a\}}$.

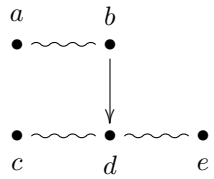


Figure 7.1: Firing a resolves the conflict between c and e, separating them into two disjoint stopping prefixes. The configuration $\{a, c\}$ is therefore recursively stopped, but not stopped. Adapted from [2].

As a final preliminary for branching cells, we define an *initial stopping prefix* to represent a prefix that cannot be made smaller while remaining a stopping prefix.

Definition 7.3 (Initial stopping prefix)**.** A stopping prefix B is said to be *initial* if $\emptyset$ is the only stopping prefix strictly contained in B.

Definition 7.4 (Branching cell)**.** For $v \in \mathcal{W}_{\mathcal{E}}$ a recursively stopped configuration, an initial stopping prefix of $\mathcal{E}^v$ is called a *branching cell*. We denote with $\delta(v)$ the set of branching cells that are initial prefixes of $\mathcal{E}^v$.

A branching cell is therefore an initial stopping prefix of the event structure that is still left after firing the configuration v. This means that it is dynamically determined during the execution of the event structure.

In contrast with the cells from Definition 6.4, branching cells are only defined as prefixes of a future of an event structure. Furthermore, branching cells are always closed under conflict by definition of a stopping prefix.

7.2 Locally Randomised Event Structures

To be able to use branching cells to attach probabilities to event structures, a generalisation of the probabilistic event structure with independence from Definition 6.3 is needed.

The main difference between this approach and the approach with independence is that this approach is only concerned with making sure that the probabilities for the maximal configurations are correct. This allows us to define the probability on the choices in branching cells instead of cells and deal with confusion correctly.

The analogue of the covering (see Definition 6.1) in this model is the *shadow*. The shadow of a configuration v represents the set of maximal configurations that are compatible with v.

Definition 7.5 (Shadow)**.** Let $\mathcal{E}$ be an event structure and v a configuration of $\mathcal{E}$. We define the shadow of v as a subset $\mathcal{S}(v)$ of the set of maximal configurations $\Omega_{\mathcal{E}}$, given by

$$\mathcal{S}(v) \coloneqq \{\omega \in \Omega_{\mathcal{E}} \mid v \subseteq \omega\}.$$

Let $\mathfrak{S}_{\mathcal{E}}$ be σ-algebra of $\Omega_{\mathcal{E}}$, defined as the σ-algebra generated by the set of shadows $\mathcal{S}(v)$, with v ranging over the finite configurations of $\mathcal{E}$.

Definition 7.6 (Probabilistic event structure)**.** A *probabilistic event structure* is a pair $(\mathcal{E}, \mathbb{P})$ where $\mathcal{E}$ is an event structure and $\mathbb{P}$ is a probability on the space $(\Omega_{\mathcal{E}}, \mathfrak{S}_{\mathcal{E}})$.

Since the probabilities are only defined for the maximal configurations, we define a *likelihood* for the other configurations.

Definition 7.7 (Likelihood)**.** For a probabilistic event structure, the likelihood of $\mathbb{P}$ is defined as the real-valued function $q : \mathcal{V}_{\mathcal{E}} \to \mathbb{R}$ defined as

$$\forall v \in \mathcal{V}_{\mathcal{E}} \qquad q(v) = \mathbb{P}(\mathcal{S}(v)),$$

where $\mathcal{V}_{\mathcal{E}}$ is the set of configurations of $\mathcal{E}$ as usual.

Now we can define *locally randomised event structures*, where probabilities are attached to the maximal configurations of each branching cell. This probability is local and therefore analogous to a cell valuation (see Definition 6.5).

Definition 7.8 (Locally randomized event structure)**.** A *locally randomised event structure* is a pair $\left(\mathcal{E}, (p_x)_{x \in X}\right)$, where X is the set of branching cells of $\mathcal{E}$, and for each $x \in X$, p_x is a probability over $(\Omega_x, \mathfrak{S}_x)$, the maximal configurations in the sub-event structure induced by x.

To connect the locally randomised event structure to the probabilistic event structure, we define the *covering* map. This map represents the set of branching cells that are part of the execution of a configuration. Note that this map serves a different purpose than the covering defined in the previous chapter (see Definition 6.1).

Definition 7.9 (Covering)**.** Call the covering map of an event structure $\mathcal{E}$, the map $\Delta(v)$ for $v \in \mathcal{V}_{\mathcal{E}}$ and

$$\begin{aligned} \Delta(v) &\coloneqq \overline{\Delta}(v) \setminus \delta(v), \\ \overline{\Delta}(v) &\coloneqq \{x \in \delta(v') \mid v' \in \mathcal{W}, v' \subseteq v\}. \end{aligned}$$

Theorem 7.1 (Abbes and Benveniste [2, Thm. 5])**.** *Let $\left(\mathcal{E}, (p_x)_{x \in X}\right)$ be a locally randomised event structure. Then there exists a unique probabilistic event structure $(\mathcal{E}, \mathbb{P})$ such that, for every stopping prefix B:*

$$\forall v \in \Omega_B, \quad \mathbb{P}(\mathcal{S}(v)) = \prod_{x \in \Delta(v)} p_x(v \cap x).$$

Theorem 7.1 is a generalisation of Proposition 6.2 to event structures with confusion.

To construct a locally randomised event structure from a probabilistic event structure, we construct the probabilities $(p_x)_{x\in X}$ from $\mathbb{P}$. Given a branching cell $x \in X$ and $\omega_x \in \Omega_x$, we do this as follows:

$$p_x(\omega_x) \coloneqq \frac{\mathbb{P}\big(\{\omega \in \Omega_{\mathcal{E}} \mid x \in \overline{\Delta}(\omega),\ \omega \cap x = \omega_x\}\big)}{\mathbb{P}\big(\{\omega \in \Omega_{\mathcal{E}} \mid x \in \overline{\Delta}(\omega)\}\big)}. \tag{7.1}$$

However, this definition is only correct when $\mathbb{P}$ is a *distributed probability*.

Definition 7.10 (Thin prefix)**.** Let ξ be a subset of $\delta(\emptyset)$. We call a prefix B of $\mathcal{E}$ *thin* if it is of the form $\bigcup_{x\in\xi} x$. We denote a thin prefix as B_ξ.

Definition 7.11 (Probabilistic future)**.** For a finite configuration v, the *probabilistic future* $(\mathcal{E}^v, \mathbb{P}^v)$ is defined as

$$\mathbb{P}^v(\cdot) \coloneqq \frac{1}{q(v)}\,\mathbb{P}(\cdot),$$

with $q(v) = \mathbb{P}(\mathcal{S}(v))$ as in Definition 7.7. The likelihood associated with this probabilistic future is

$$q^v(\omega) = \frac{q(v \cup \omega)}{q(v)},$$

where ω is a finite configuration of $\mathcal{E}^v$.

Definition 7.12 (Distributed probability)**.** We call a probability $\mathbb{P}$ *distributed* if for each recursively stopped configuration v and each thin prefix B^v_ξ in $\mathcal{E}^v$, we have

$$\forall \omega \in \Omega_{B^v_\xi} \qquad q^v(\omega) = \prod_{x\in\xi} p_x(\omega \cap x).$$

Theorem 7.2 (Abbes [1, Ch. 4, Thm. IV-2.2])**.** *Every probabilistic event structure $(\mathcal{E}, \mathbb{P})$ with a distributed probability induces a locally randomised event structure with probability $(p_x)_{x\in X}$ by Equation* (7.1)*. Applying Theorem 7.1 then gives back $\mathbb{P}$.*

This theorem is a counterpart of Proposition 6.3 for the branching cell approach. Theorems 7.1 and 7.2 imply that probabilistic event structures with a distributed probability and locally randomised event structures are equivalent.

7.3 Markov Nets

So far, we focused on probabilistic event structures. In this section we will briefly discuss the framework presented in this chapter can be used to randomise Petri nets.

Abbes and Benveniste [2] have proposed to call the randomised Petri nets obtained from locally randomised event structures *Markov nets* since they provide a concurrent generalisation of discrete Markov chains. However, the focus here is to treat Markov nets only in as far as they are relevant to randomisation of Petri nets.

The general approach of Markov nets is to map a locally finite Petri net N to an event structure and attach a probability distribution to each of the isomorphism classes of the branching cells, called *dynamic clusters*. The probabilities on the isomorphism classes then act as probabilities on the execution of the Markov net.

Definition 7.13 (Dynamic cluster)**.** An isomorphism class of branching cells is called a dynamic cluster of N. We denote by Σ the (finite) set of dynamic clusters. Dynamic clusters are denoted by the boldface symbol $\mathbf{s}$.

Definition 7.14 (Markov net)**.** A Markov net is a pair $\left(N, (p_{\mathbf{s}})_{\mathbf{s}\in\Sigma}\right)$, where N is a locally finite 1-safe Petri net and $p_{\mathbf{s}}$ is a probability on the finite set $\Omega_{\mathbf{s}}$ for every $\mathbf{s} \in \Sigma$.

By doing this, confusion is not removed from the net, but rather the execution of the net is governed by the probability distributions on the dynamic clusters. The drawback of this approach is that the execution of the net must be computed dynamically during the execution of the net and is dependent on the current marking.

Chapter 8

Structural Branching Cells Method

A novel method of removing confusion was introduced by Bruni, Melgratti, and Montanari [5], which we will refer to as the s-cells method. The difference with the method by Abbes and Benveniste is that this method is static rather than dynamic: instead of altering the execution order by computing event structures, the net itself is altered.

This chapter first discusses the preliminaries for the construction in Sections 8.1 to 8.4. The construction itself is detailed in Section 8.5. Finally, randomisation of the final net is explored in Section 8.6. This chapter follows the definitions and notation in [5], except for the section on dependence and the details on pruning.

8.1 Persistent Places

Persistent places are places that stay marked forever when they are initially marked. In the construction, these places will represent the fact that a corresponding place cannot be marked. The persistency is necessary because the fact that a place will not be marked will hold true for the rest of the execution.

To accommodate for persistency, there must be a distinction between normal places and persistent places. Therefore, the set of places is split into the set of normal places P and the set of persistent places $\mathbf{P}$. The Petri net is then given by $(\mathbb{S}, T, F)$, where $\mathbb{S} := P \cup \mathbf{P}$. The definition of a marked net is extended such that the marking is a multiset $m \in \mathbb{N}_\infty^P$ with $\forall p \in P \;\; m(p) \in \mathbb{N}$ and $\forall \mathbf{p} \in \mathbf{P} \;\; m(\mathbf{p}) \in \{0, \infty\}$. To ensure that each persistent place is filled with ∞ when it is marked, the postset of a transition t is a multiset with $(t^\bullet)(\mathbf{p}) = \infty$ if $(t, \mathbf{p}) \in F$ for any $\mathbf{p} \in \mathbf{P}$.

A marked Petri net (P, T, F, m_0) is said to be 1-∞-*safe* if all reachable markings m satisfy that $m(p) \in \{0, 1\}$ for all $p \in P$ and $m(\mathbf{p}) \in \{0, \infty\}$ for all $\mathbf{p} \in \mathbf{P}$.

Graphically, persistent places will be represented by having a double border and they will contain a single token if $m(\mathbf{p}) = \infty$.

8.2 Structural Branching Cells

The s-cells method uses a variation on the branching cells by Abbes and Benveniste, called *structural branching cells* or *s-cells*. Intuitively, an s-cell represents a locus of choice: a set of transitions that form mutually exclusive firing sequences. Bruni, Melgratti, and Montanari [5], define s-cells with the help of three new relations:

$$\begin{aligned} \mathsf{Pre} &\coloneqq F \cap (P \times T), \\ \sqsubseteq &\coloneqq \left(\preceq \cup \mathsf{Pre}^{-1}\right)^*, \\ \leftrightarrow &\coloneqq \{(x, y) \mid x \sqsubseteq y \wedge y \sqsubseteq x\}. \end{aligned}$$

Definition 8.1 (S-cell)**.** Given a finite occurrence net $N = (P, T, F)$, the s-cell $[t]$ assigned to transition t is the equivalence class of t with respect to the equivalence relation $\leftrightarrow$. Given an s-cell $\mathbb{C}$ of N, we denote with $N_{\mathbb{C}}$ the net N restricted to $\mathbb{C}$ such that the nodes are in $\mathbb{C} \cup \bigcup_{t \in \mathbb{C}} t^\bullet$. We write ${}^\circ\mathbb{C}$ and $\mathbb{C}^\circ$ for ${}^\circ N_{\mathbb{C}}$ and $N^\circ_{\mathbb{C}}$ respectively. The set of s-cells in a Petri net is given by $\textsc{bc}(N) \coloneqq \{[t] \mid t \in T\}$.

A transaction θ of $N_{\mathbb{C}}$ is denoted $\theta : \mathbb{C}$. A transaction θ is uniquely determined by its set of transitions. Therefore, we will identify transactions with the set of transitions.

To see how this definition works, consider the net in Figure 8.1. Each dashed region in this figure represents an s-cell. To determine the s-cell $\mathbb{C}_1 = [t_3]$, we first consider the place c in the preset of t_3. Since $(c, t_3) \in \mathsf{Pre}$, we conclude that $(t_3, c) \in \mathsf{Pre}^{-1}$, which implies by definition of $\sqsubset$ that $t_3 \sqsubseteq c$. Additionally, we see that $c \preceq t_3$, therefore $c \sqsubseteq t_3$. Finally, we get that $t_3 \leftrightarrow c$. So c belongs to $\mathbb{C}_1$. Because $\sqsubseteq$ is defined as the transitive closure of $\preceq \cup \mathsf{Pre}^{-1}$, a similar argument is made for the inclusion of t_4, by noting that $c \preceq t_4$ and $(t_4, c) \in \mathsf{Pre}^{-1}$. Using the same reasoning, the place d and transition t_5 are added are added to $\mathbb{C}_1$ as well. Although $t_1 \preceq d$, the transition t_1 is not added to the net since $(d, t_1) \notin \mathsf{Pre}^{-1}$.

Note that s-cells are closed under direct conflict, since $\#_0 \subseteq \leftrightarrow$. Additionally, s-cells contain transitions that are causally dependent on each other. Consider $\mathbb{C}_2$ in Figure 8.1. This s-cell must be considered a single locus of choice, since the choices for the places b and f are dependent on each other. If t_2 is fired first, t_6 becomes enabled and a choice between t_6 and t_7 must be made. Conversely, if t_7 is fired first, a choice must be made between t_2 and t_8.

Starting the computation of the s-cell from t_1, the same arguments as before apply to conclude that

$$\{t_1, t_2, t_8, a, b, g\} \subseteq [t_1].$$

The fact that the place e belongs to this s-cell follows from

$$t_2 \preceq e \quad \text{and} \quad e \preceq t_6 \wedge (t_6, f) \in \mathsf{Pre}^{-1} \wedge f \preceq t_8 \wedge (t_8, b)\, \mathsf{Pre}^{-1} \wedge b \preceq t_2,$$

implying $t_2 \sqsubseteq e$ and $e \sqsubseteq t_2$, respectively. Repeating the same argument for the other nodes gives

$$[t_1] = \{t_1, t_2, t_6, t_7, t_8, a, b, e, f, g\}.$$

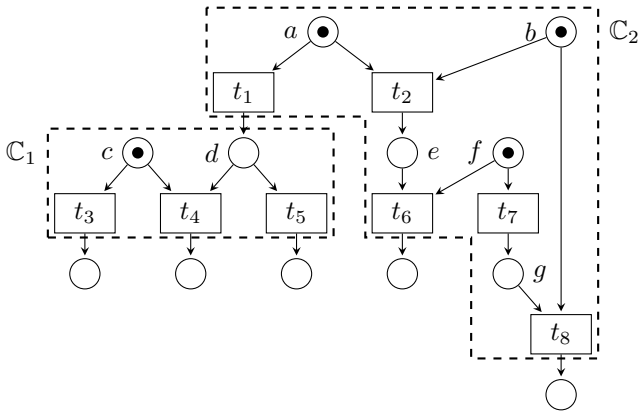


Figure 8.1: A Petri net with two s-cells represented by regions with a dashed border.

From a graph theoretic perspective, s-cells can be defined as strongly connected components of a directed graph. Indeed, consider the graph $G = (P \cup T, F \cup \mathsf{Pre}^{-1})$. From elementary set theory, we have

$$\left(\preceq \cup \mathsf{Pre}^{-1}\right)^* = \left(F \cup \mathsf{Pre}^{-1}\right)^*,$$

and we further note that the existence of a path in a graph with edges $F \cup \mathsf{Pre}^{-1}$ is equivalent to the transitive closure of $F \cup \mathsf{Pre}^{-1}$. Therefore, the strongly connected components of G are the s-cells. This connection is important because it shows that established algorithms for computing strongly connected components, such as the algorithm by Tarjan [11], can be used to identify s-cells.

8.3 Dynamic Petri Nets

The s-cells method also relies on the concept of dynamic Petri nets. These are nets that can grow during the execution. Conceptually, each transition t is given a (possibly empty) set of transitions that are added to the net when t is fired. Dynamic Petri nets are not strictly necessary for the construction, but they are used as a convenient intermediate step.

Figure 8.2 shows an example of a dynamic Petri net. Initially, the transition t_2 cannot be fired (Figure 8.2a), indicated by the dashed border. Only after t_3 is fired, t_3 is activated and can be fired (Figure 8.2b).

Definition 8.2 (Dynamic transition)**.** A dynamic transition is a pair

$$t = (S, D),$$

(a) The initial net D_0. The transition t_2 is not initially available to be fired.

(b) The net D_1 after firing t_3. The transition t_2 has been activated.

Figure 8.2: The execution of a dynamic transition in a dynamic Petri net. Transitions that have not been added to the net yet are shown with a dashed border and can not be fired. The arrow between the transitions t_3 and t_2 represents the fact that t_3 activates t_2.

where S is a set of places called the preset and $D = (T, m)$ is called the postset. Here, T should be interpreted as the set of transitions added to the net and $m \in \mathbb{N}_\infty^{\mathbb{S}}$ as the multiset of tokens added to the marking of the net after firing t, which is analogous to the postset in a static Petri net. In this case, we say that t *activates* the transitions in T. We alternatively write the transition t as

$$t = S \to D.$$

As with static transitions, we denote the preset S and postset D of a dynamic transition with ${}^\bullet t$ and $t^\bullet$, respectively.

Definition 8.3 (Dynamic Petri net)**.** A dynamic Petri net is defined as a pair $D = (T, m)$, where T is the set of currently available dynamic transitions and $m \in \mathbb{N}_\infty^{\mathbb{S}}$ is the marking. For every $(S, D') \in T$, the postset D' must also be a dynamic Petri net.

Given a dynamic Petri net $D = (T, b)$, a dynamic transition $t = S \to (T_t, m_t) \in T$ is said to be enabled if $m_t \subseteq m$. When the transition is fired, the result is a new dynamic Petri net $N' = (T', m')$, with $T' = T \cup T_t$ and $m' = (m \setminus S) + m_t$.

Definition 8.4 (From static to dynamic)**.** A normal, static transition t_{static} with ${}^\bullet t_{\text{static}} = m_1$ and $t_{\text{static}}^\bullet = m_2$ is equivalent to the transition $m_1 \to (\emptyset, m_2)$ in a dynamic Petri net. For a Petri net $N = (P, T, F, m)$, we therefore define an equivalent dynamic Petri net

$$\textsc{dyn}(N) = (\{{}^\bullet t \to (\emptyset, t^\bullet) \mid t \in T\}, m).$$

Consider the dynamic Petri net in Figure 8.2, where transitions with a dashed border are not activated. The transitions in this net are:

$$\begin{aligned} t_1 &= \{a\} \to (\emptyset, \{c\}), \\ t_2 &= \{a\} \to (\emptyset, \{d\}), \\ t_3 &= \{b\} \to (\{t_2\}, \{e\}). \end{aligned}$$

The initial net, shown in Figure 8.2a is given by

$$D_0 = (T_0, m_0) = (\{t_1, t_3\}, \{a, b\}).$$

After firing t_3, the transition t_2 is activated, and the net becomes

$$\begin{aligned} D_1 &= (T_0 \cup \{t_2\}, ((m_0 \setminus \{b\}) \cup \{e\})) \\ &= (\{t_1, t_2, t_3\}, \{a, e\}), \end{aligned}$$

which is shown in Figure 8.2b. The other transitions can then be fired like they could in a normal Petri net.

Using the fact that the postset of each dynamic transition must be a dynamic Petri net according to Definition 8.3, the set of dynamic Petri nets over the set of places $\mathbb{S}$ is defined below.

Definition 8.5 (Set of dynamic Petri nets)**.** The set $\textsc{dn}(\mathbb{S})$ is defined as

$$\begin{aligned} \textsc{dn}_0(\mathbb{S}) &= \{\textsc{dyn}(N) \mid N \text{ is a Petri net with places } \mathbb{S}\}, \\ \textsc{dn}_{n+1}(\mathbb{S}) &= \left\{(T, m) \;\middle|\; T \subseteq 2^{\mathbb{S}} \times \textsc{dn}_n(\mathbb{S}) \wedge T \text{ finite} \wedge m \in \mathbb{N}_\infty^{\mathbb{S}}\right\}, \\ \textsc{dn}(\mathbb{S}) &= \bigcup_{n \in \mathbb{N}} \textsc{dn}_n(\mathbb{S}). \end{aligned}$$

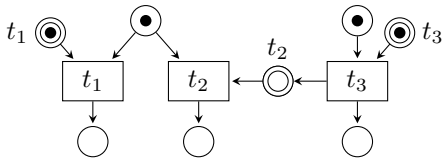
A dynamic Petri net is converted into a static Petri net using persistent places. The intuition is that whenever a transition is activated by another, a persistent place is put in between these transitions. More precisely, a persistent place is added representing whether a transition is activated. However, the persistent places attached to transitions that are initially activated in the dynamic net can be removed or "pruned" without changing the firing sequences of the net. For example, the static net corresponding to the dynamic net in Figure 8.2 is shown in Figure 8.3a, along with the pruned version in Figure 8.3b.

Definition 8.6 (From dynamic to static)**.** Given $D = (T, b) \in \textsc{dn}(\mathbb{S})$, the corresponding p-net $(\!|D|\!)$ is defined as

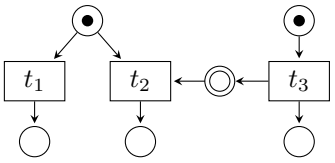
$$(\!|D|\!) \coloneqq \big(\mathbb{S} \cup \mathbf{P}_{\mathbb{T}(D)},\ \mathbb{T}(D),\ F,\ b \cup b_T\big),$$

where

- $\mathbb{T}(D) = T \cup \bigcup_{t \in T} \mathbb{T}(t^\bullet)$ is the set of transitions in the dynamic net;
- $\mathbf{P}_{\mathbb{T}(D)} = \{\mathbf{p}_t \mid t \in \mathbb{T}(D)\}$ is a set containing a persistent place for each transition; and
- F is such that for any $t = S \to (T', b') \in \mathbb{T}(D)$ we have $t : {}^\bullet t \cup \{\mathbf{p}_t\} \to b' \cup b_{T'}$.



(a) The static Petri net corresponding to the dynamic net in Figure 8.2.



(b) The pruned version of the static net in Figure 8.3a.

Figure 8.3: Static net corresponding to the dynamic net in Figure 8.2.

8.4 Dependence

The s-cells method additionally uses the $\ominus$ operation. Given a Petri net N and a place p, the expression $N \ominus p$ represents the net N without the place p and the nodes that are causally dependent on it. We define this operation with a binary relation representing *dependence* between the nodes of a Petri net.

Definition 8.7 (Dependence)**.** We define the relation $\lhd$ by recursion as follows:

$$\begin{aligned} \lhd_0 &\coloneqq \mathrm{Id} \\ \lhd_{n+1} &\coloneqq \lhd_n \cup \{(p,x) \in P \times (P \cup T) \mid {}^\bullet p \neq \emptyset \wedge \forall t \in {}^\bullet p \;\; t \lhd_n x\} \\ &\qquad \cup \{(t,x) \in T \times (P \cup T) \mid \exists p \in {}^\bullet t \;\; p \lhd_n x\} \\ \lhd &\coloneqq \bigcup_{n \in \mathbb{N}} \lhd_n, \end{aligned}$$

where $\mathrm{Id} = \{(x,x) \mid x \in P \cup T\}$. Given nodes $x, y \in P \cup T$, we say that x is dependent on y if $x \lhd y$.

Note that the condition ${}^\bullet p \neq \emptyset$ is added to account for the vacuous truth of universal quantification, since places should not be dependent on nodes they are not connected to.

If a transition t is dependent on p, then t can only be fired if p is marked at some point during the execution of the net. It can therefore not be fired when p is removed from the net. If a place q is dependent on p it can only be marked if p is marked at some point during the execution, meaning that the removal of p would make it impossible to mark q.

Definition 8.8 (Removal of dependence)**.** We denote with $N \ominus p$ the net N with the nodes dependent on the place $p \in P$ removed. This is given by

$$N \ominus p \coloneqq N \setminus \{n \in P \cup T \mid n \lhd p\}.$$

Proposition 8.1. *For a finite acyclic Petri net N and a place $p \in {}^\circ N$, the net $N \ominus p$ is the smallest subnet of N closed under the following rules:*

$$\frac{q \in {}^\circ N \setminus \{p\}}{q \in N \ominus p}, \qquad \frac{t \in N \quad {}^\bullet t \subseteq N \ominus p}{t \in N \ominus p}, \qquad \frac{t \in N \ominus p \quad q \in t^\bullet}{q \in N \ominus p}.$$

In particular, our definition of $\ominus$ coincides with the definition given in [5].

Proof. First, we prove that $N \ominus p$ is closed under the rules. Let $q \in {}^\circ N \setminus \{p\}$ be an initial place. By definition of ${}^\circ N$, the preset of q is then empty. Therefore, we have that $\neg\exists n \in \mathbb{N}\ ({}^\bullet q \neq \emptyset \wedge \forall t \in {}^\bullet q\ \ t \lhd_n p)$. By definition of $\lhd$ this implies that $\neg(q \lhd p)$ and we conclude that $q \in N \ominus p$.

Now let $t \in N$ be a transition and ${}^\bullet t \subseteq N \ominus p$. This implies $\forall q \in {}^\bullet t\ \ q \in N \ominus p$. Hence, we have that $\forall q \in {}^\bullet t\ \ \neg(q \lhd p)$. Since $\lhd = \cup_{n\in\mathbb{N}} \lhd_n$, this implies $\forall q \in {}^\bullet t\ \ \neg\exists n \in \mathbb{N}\ \ q \lhd_n p$. Rewriting this gives $\neg\exists n \in \mathbb{N}\ \ \exists q \in {}^\bullet t\ \ q \lhd_n p$. Therefore, there is no n such that $\exists q \in {}^\bullet t\ \ q \lhd_n p$ holds. So by definition of $\lhd$, we have $\neg(t \lhd p)$ and $t \in N \ominus p$.

Now let $q \in N \setminus {}^\circ N$ be a non-initial place with a $t \in {}^\bullet q$ such that $t \in N \ominus p$. This implies that $\neg(t \lhd p)$ and, again by definition of $\lhd$, we have $\neg\exists n \in \mathbb{N}\ \ t \lhd_n p$. Hence, we have that $\neg\exists n \in \mathbb{N}\ \ \forall t \in {}^\bullet q\ \ t \lhd_n p$. Therefore, there is no n such that ${}^\bullet p \neq \emptyset \wedge \forall t \in {}^\bullet q\ \ t \lhd_n p$ holds and we have $\neg(q \lhd p)$. Finally, this implies $q \in N \ominus p$.

We conclude that $N \ominus p$ is closed under the rules.

Now we show that if $N' \subseteq N$ is a Petri net closed under the rules then $N \ominus p \subseteq N'$. We assume towards a contradiction that there is some x_0 such that $x_0 \in N \ominus p$ and $x_0 \notin N'$. We will construct an infinite path in the directed graph induced by the flow relation.

First note that $x_o \notin {}^\circ N$, since $p \notin N\ominus$ and ${}^\circ N \setminus \{p\} \subseteq N'$ because N' is closed under the rules. We then have two cases: x_0 is a non-initial place or a transition.

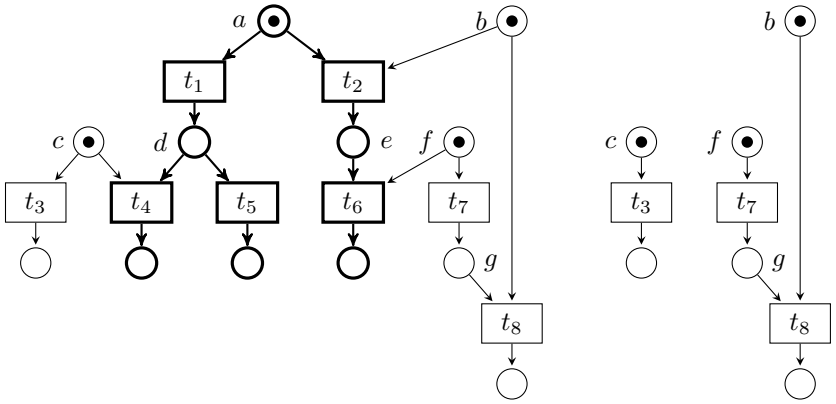
Consider the first case. Since N' is closed under the rules, we have that $\forall t \in {}^\bullet x_0\ \ t \notin N'$. Furthermore, if we assume towards a contradiction that $\forall t \in {}^\bullet x_0\ \ t \notin N \ominus p$, then we get that $\forall t \in {}^\bullet x_0\ \ t \lhd p$ and $\forall t \in {}^\bullet x_0\ \ \exists n \in \mathbb{N}\ \ t \lhd_n p$. Since N is finite, ${}^\bullet x$ is finite. Therefore, there is an $n \in \mathbb{N}$ such that $\forall t \in {}^\bullet x_0\ \ t \lhd_n p$. Furthermore, since $x_0 \notin {}^\circ N$, we have that ${}^\bullet x_0 \neq \emptyset$. Therefore, there is an $n \in \mathbb{N}$ where the condition ${}^\bullet x_0 \neq \emptyset \wedge \forall t \in {}^\bullet x_0\ \ t \lhd_n p$ holds and we have that $x_0 \lhd p$ and $x_0 \notin N \ominus p$, contradicting the assumption that $x_0 \in N \ominus p$. Therefore we conclude that $\exists t \in {}^\bullet x_0\ \ t \in N \ominus p$. We let x_1 be a transition such that $x_1 \in N \ominus p$ and $x_1 \notin N'$.

In the second case, we must have that ${}^\bullet x_0 \not\subseteq N'$ since N is closed under the rules. We assume towards a contradiction that ${}^\bullet x_0 \not\subseteq N \ominus p$. We get $\exists q \in {}^\bullet x_0\ \ q \notin N \ominus p$, which by definition of $\ominus$ implies that $\exists q \in {}^\bullet x_0\ \ q \lhd n$. By definition of $\lhd$ we then have that $\exists q \in {}^\bullet x_0\ \ \exists n \in \mathbb{N}\ \ q \lhd_n p$. Rewriting then gives $\exists n \in \mathbb{N}\ \ \exists q \in {}^\bullet x_0\ \ q \lhd_n p$. Since there is an $n \in \mathbb{N}$ such that $\exists q \in {}^\bullet x_0\ \ q \lhd_n p$ we conclude that $x_0 \lhd p$ and $x_0 \notin N \ominus p$, contradicting the assumption that $x_0 \in N \ominus p$. Therefore, we conclude that ${}^\bullet x_0 \subseteq N \ominus p$. We let x_1 be a place such that $x_1 \in N \ominus p$ and $x_1 \notin N'$.

In both cases, there must be some $x_1 \in {}^\bullet x_0$ such that $x_1 \in N \ominus p$ and $x_1 \notin N'$. Repeating this construction then gives a sequence $x_1, x_2, \ldots$. Since N is finite, there must be $x_i = x_j$ with $i \neq j$ in this sequence. Since $x_{n+1} \in {}^\bullet x_n$ for all $n \in \mathbb{N}$, we have by definition of the preset that $(x_{n+1}, x_n) \in F$. The sequence $x_i, \ldots, x_j$ would therefore be a cycle in the graph induced by the flow relation, but this is not possible since N is acyclic. Therefore we have arrived at a contradiction and conclude that $x_0 \in N \ominus' p$. Then we have that $N \ominus p \subseteq N'$.

This shows that $N \ominus p$ is the smallest subnet of N closed under the rules. □

As an example of dependence, consider again the net $N = (P, T, F)$ in Figure 8.1. A transition such as t_4 is dependent on a if one of the places in its preset is dependent on a. The same does not hold for places, which are only dependent on a if all transitions in the preset are dependent on a. For this net, the removal of the nodes dependent on a give the net $N \ominus a$, shown in Figure 8.4b.



(a) The net from Figure 8.1 with all nodes $x \in P \cup T$ dependent on a shown in bold.

(b) The net $N \ominus a$.

The definition of $\ominus$ above could be a first step in generalising the s-cells method to locally finite Petri nets. Consider for example the locally finite net N in Figure 8.5. Following the definition of $\ominus$ in [5], $N \ominus b$ would remove the entire net even though t_1 can still be fired if b is removed from the net. However, following our definition, $N \ominus b$ only removes b, t_2 and c.

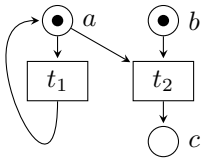


Figure 8.5: A locally finite Petri net

8.5 The Construction

With the definitions of s-cells and dynamic Petri nets in place, the construction proposed by Bruni, Melgratti, and Montanari [5] can be explored. The final construction consists of 3 steps:

1. first, a confusion-free dynamic Petri net is constructed using the s-cells;

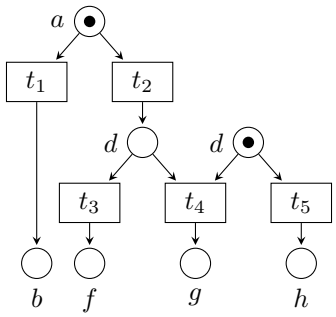


Figure 8.6: The running example for this section.

2. then, a confusion-free static Petri net constructed from the previously obtained dynamic Petri net;

3. finally, the static net is pruned to simplify it.

In this section, the places that represent the "negative" information in the net will be denoted by a bar: $\bar{\mathbf{p}}$[1]. Additionally, these places are typeset in bold since they are persistent. The notation is extended for sets of places, so that for a set P of places, we denote by $\overline{P}$ the set $\{\bar{p} \mid p \in P\}$.

For the rest of this section we assume that the input of the construction is a finite occurrence net $N = (P, T, F, m)$.

Running example A running example will be used throughout this section to illustrate the steps of the construction. The example is shown in Figure 8.6. Note that the example has both asymmetric and symmetric confusion.

Step 1 From the input net N, a dynamic net $[\![N]\!]$ is constructed. The operation $[\![\cdot]\!]$ is defined as follows.

Definition 8.9 (From s-cells to dynamic Petri nets)**.** Let $N = (P, T, F, m)$ be a marked occurrence net. Its dynamic p-net $[\![N]\!] \in \text{DN}(P \cup \overline{\mathbf{P}})$ is defined as

$$[\![N]\!] = (T_{\text{pos}} \cup T_{\text{neg}}, m)$$

where

$$T_{\text{pos}} = \left\{ {}^\circ\mathbb{C} \to \left(\emptyset, \theta^\circ \cup \overline{\mathbb{C}^\circ \setminus \theta^\circ}\right) \;\middle|\; \mathbb{C} \in \text{BC}(N) \text{ and } \theta : \mathbb{C} \right\}$$

$$T_{\text{neg}} = \left\{ \bar{\mathbf{p}} \to \left(T', \overline{\mathbb{C}^\circ \setminus (N_{\mathbb{C}} \ominus p)^\circ}\right) \;\middle|\; \mathbb{C} \in \text{BC}(N) \text{ and } p \in {}^\circ\mathbb{C} \text{ and } (T', b) = [\![N_{\mathbb{C}} \ominus p]\!] \right\}$$

[1] Note that the bar does not denote set complement

The purpose of T_{pos} is to add the flow of the s-cell to the new confusion-free net. This means that a transition is created for every transaction θ in the cell, with the initial places of the cell as its preset and the places marked at the end of the transaction as its postset. Additionally, the negative places of the places that are not marked at the end of the transaction are added to the postset. These negative places will always be persistent.

The set T_{neg} then further connects the negative places to each other. For each initial place p of $\mathbb{C}$, we determine what final places of $\mathbb{C}$ cannot be reached if the place p is never marked, by recursively calling $[\![N_{\mathbb{C}} \ominus p]\!]$. When $N_{\mathbb{C}} \ominus p$ is empty, then the absence of a token at p makes it impossible to fire the rest of the cell, so the place has to be connected to all places in $\mathbb{C}^\circ$. When $\mathbb{C}$ is not empty, there is a subcell inside of the current cell which can be executed without ever marking p. Therefore, $\bar{\mathbf{p}}$ should be connected to each transaction in $[\![N_{\mathbb{C}} \ominus p]\!]$. Bruni, Melgratti, and Montanari [5, Cor. 3.13] have shown that any net $[\![N]\!] \in \text{DN}\big(P \cup \overline{\mathbf{P}}\big)$ is confusion-free.

Running example The s-cells of the running example are shown in Figure 8.7. We let $\mathbb{C}_1 \coloneqq [t_1]$ and $\mathbb{C}_2 \coloneqq [t_3]$. We can treat every s-cell independently and combine the dynamic nets by taking the union of the set of transitions.

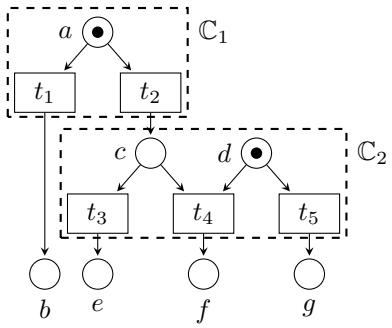


Figure 8.7: The s-cells $\mathbb{C}_1$ and $\mathbb{C}_2$ in the running example.

For the positive transitions from $\mathbb{C}_1$, we determine the transactions of $\mathbb{C}_1$: $\{t_1\}$ and $\{t_2\}$. For each of these transactions $\theta : \mathbb{C}_1$ a transition

$$^\circ\mathbb{C}_1 \to \Big(\emptyset, \theta^\circ \cup \overline{\mathbb{C}_1^\circ \setminus \theta^\circ}\Big)$$

is created, such that

$$\begin{aligned} \{t_1\} \quad &\text{gives} \quad \{a\} \to (\emptyset, \{b\} \cup \{\bar{\mathbf{c}}\}), \\ \{t_2\} \quad &\text{gives} \quad \{a\} \to \big(\emptyset, \{c\} \cup \{\bar{\mathbf{b}}\}\big). \end{aligned}$$

The resulting Petri net is shown in Figure 8.8a. Similarly, for the s-cell $\mathbb{C}_2$, the transactions are $\{t_4\}$ and $\{t_3, t_5\}$. Therefore, the transitions in the net are

$$\{a, d\} \to (\emptyset, \{f\} \cup \{\bar{\mathbf{e}}, \bar{\mathbf{g}}\}) \quad \text{and} \quad \{a, d\} \to \big(\emptyset, \{e, g\} \cup \{\bar{\mathbf{f}}\}\big).$$

This results in the Petri net in Figure 8.9a.

The negative connections in $\mathbb{C}_1$ are simple as well. Since $\{a\} = {}^\circ\mathbb{C}_1$, the Petri net $N_{\mathbb{C}_1} \ominus a$ is empty and therefore $[\![N_{\mathbb{C}_1} \ominus a]\!] = (\emptyset, \emptyset)$. The transition associated to $\bar{\mathbf{a}}$ is then

$$\{\bar{\mathbf{a}}\} \to \big(\emptyset, \{\bar{\mathbf{b}}, \bar{\mathbf{c}}\}\big).$$

Therefore, we get the Petri net shown in Figure 8.8b.

For $\mathbb{C}_2$, there are two initial places of $N_{\mathbb{C}_2}$: c and d. For each of these places, a transition is created. In the case of $\mathbb{C}_2$, the dynamic nets $[\![N_{\mathbb{C}_2} \ominus c]\!]$ and $[\![N_{\mathbb{C}_2} \ominus d]\!]$ are not empty. Consider the net $[\![N_{\mathbb{C}_2} \ominus c]\!]$, which is

$$(\{d \to (\emptyset, \{g\})\}, \emptyset).$$

The transitions in this net are then activated by the transition associated with $\bar{\mathbf{c}}$. The transition associated with $\bar{\mathbf{c}}$ therefore becomes

$$\{\bar{\mathbf{c}}\} \to \big(\{\{d\} \to (\emptyset, \{g\})\}, \{\bar{\mathbf{e}}, \bar{\mathbf{f}}\}\big).$$

Similarly, the transition associated with $\bar{\mathbf{d}}$ is

$$\{\bar{\mathbf{d}}\} \to \big(\{\{c\} \to (\emptyset, \{e\})\}, \{\bar{\mathbf{g}}, \bar{\mathbf{f}}\}\big).$$

We obtain the net shown in Figure 8.9b.

Combining the results of T_{pos} and T_{neg} for $\mathbb{C}_1$ and $\mathbb{C}_2$ then finally gives the dynamic Petri net shown in Figure 8.10.

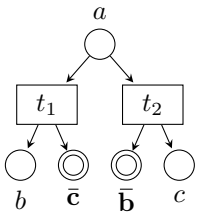
Step 2 In the second step, the dynamic Petri net is converted into a static net using Definition 8.6 giving the net $(\!|[\![N]\!]|\!)$. As shown by Bruni, Melgratti, and Montanari [5, Cor. 3.14], any Petri net $(\!|[\![N]\!]|\!)$ is confusion-free. Furthermore, for any dynamic Petri net D, the static net $(\!|N|\!)$ is 1-∞-safe if D is 1-safe, so $(\!|[\![N]\!]|\!)$ is 1-∞-safe [5, Cor. 2.5]. The Petri net $(\!|[\![N]\!]|\!)$ for the running example is shown in Figure 8.11.

Step 3 Finally, the confusion-free net $(\!|[\![N]\!]|\!)$ can be pruned. Pruning is discussed, but not formalised in [5]. Below, we attempt to define an operation that prunes as much as possible while leaving all places and transitions in the original net N. For all pruning operations in this section, we assume the input to be a marked Petri net with persistent places $N = (P \cup \mathbf{P}, T, F, m)$.

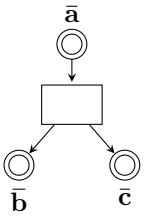
First of all, the superfluous marked persistent places resulting from the $(\!|\cdot|\!)$ operation are removed with the pruning operation

$$\textsc{prune}_{(\!|\cdot|\!)} \coloneqq N \setminus (\mathbf{P} \cap m).$$

The marked persistent places are also in ${}^\circ N$ by construction of $(\!|[\![N]\!]|\!)$. Also by construction, there are no transitions such that ${}^\bullet t \subseteq \mathbf{P} \cup m$. The result will not create confusion since it does not change the enabled transitions and the conflicts between them.

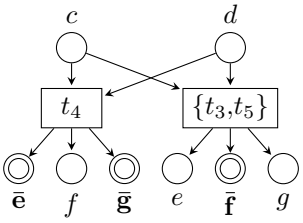


(a) T_{pos} for s-cell $\mathbb{C}_1$.

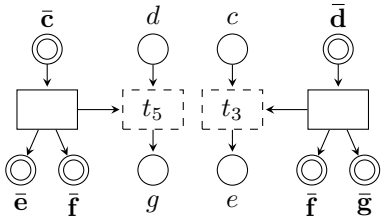


(b) T_{neg} for s-cell $\mathbb{C}_1$.

Figure 8.8: Components of $[\![N]\!]$ for the s-cell $\mathbb{C}_1$ from Figure 8.7.



(a) T_{pos} for s-cell $\mathbb{C}_2$.



(b) T_{neg} for s-cell $\mathbb{C}_2$.

Figure 8.9: Components of $[\![N]\!]$ for the s-cell $\mathbb{C}_2$ from Figure 8.7.

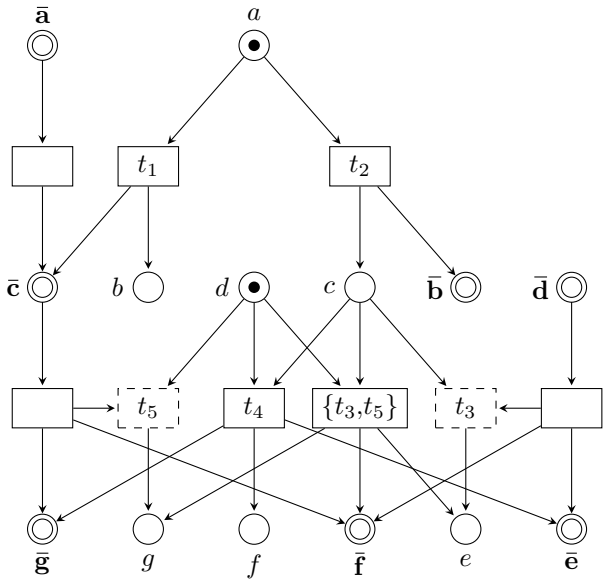


Figure 8.10: The complete dynamic Petri net $[\![N]\!]$ for the running example introduced in Figure 8.6.

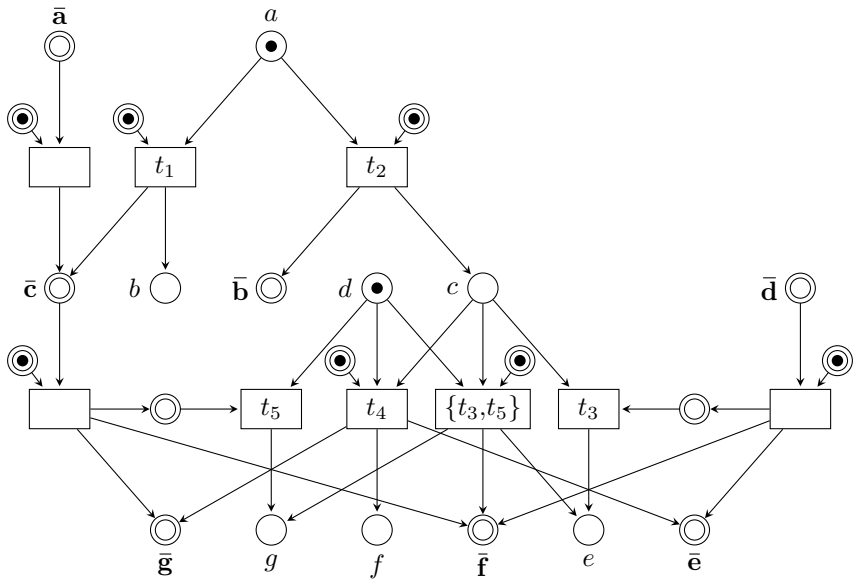


Figure 8.11: The confusion-free unpruned Petri net $(\!|[\![N]\!]|\!)$ for the running example introduced in Figure 8.6.

The second step in pruning is therefore to remove all nodes dependent on empty initial persistent places. However, we cannot simply use the intersection of $N \ominus p$ for each $p \in {}^\circ N$. To illustrate this, consider a net with two initial places p and q that are not marked. Now a third place r has two transitions in its preset ${}^\bullet r = \{t_1, t_2\}$, such that

$$t_1 \lhd p \wedge \neg(t_1 \lhd q) \wedge t_2 \lhd q \wedge \neg(t_2 \lhd p).$$

Taking the intersection $N \ominus p \cap N \ominus q$ then does not remove r, since r is in both $N \ominus p$ and $N \ominus q$, even though it cannot be marked.

Definition 8.10 (Extended dependence)**.** We define the relation $\lhd$ by recursion as follows:

$$\begin{aligned}
\lhd_o &\coloneqq \{(x, S) \mid S \subseteq P \cup T \wedge x \in S\} \\
\lhd_{n+1} &\coloneqq \lhd_n \cup \{(p, S) \mid p \in P \wedge {}^\bullet p \neq \emptyset \wedge \forall t \in {}^\bullet p \;\; t \lhd_n S\} \\
&\qquad \cup \{(s, S) \mid t \in T \wedge \exists p \in {}^\bullet t \;\; p \lhd_n S\} \\
\lhd &\coloneqq \bigcup_{n \in \mathbb{N}} \lhd_n,
\end{aligned}$$

A node $x \in P \cup T$ is said to be dependent on the set of nodes $S \subseteq P \cup T$ if $x \lhd S$.

This definition is equivalent to Definition 8.7 for $|S| = 1$. For a Petri net $N = (P \cup \overline{\mathbf{P}}, T, F)$ we then define the pruning operation

$$\textsc{prune}_\lhd(N) \coloneqq N \setminus \left\{x \in P \cup T \;\middle|\; x \lhd \left({}^\circ N \cap \overline{\mathbf{P}}\right)\right\}.$$

Since $\textsc{prune}_{\lhd}$ does not affect places that can be marked or transitions that can be fired throughout the execution of the Petri net, the firing sequences are preserved by this operation and no confusion is added.

Next, we note that final negative places, such as $\bar{\mathbf{g}}$ in Figure 8.11, do not serve a purpose. So we define a prune operation to remove all nodes that do not affect normal places as follows:

$$\textsc{prune}_{\preceq}(N) \coloneqq N \setminus \left\{x \in \overline{\mathbf{P}} \cup T \;\middle|\; \forall p \in P \;\neg(x \preceq p)\right\}.$$

The only transitions that could be removed by $\textsc{prune}_{\preceq}$are the ones that do not have a corresponding transition in the original net N, because the ones that do always have non-persistent places in their postset. This leaves the transitions shown as empty boxes in Figure 8.11. These transitions have only a single place in their preset (after the first pruning step) and they are the only transition in the postset of that place. This place is therefore also removed. Conflict is then not affected and neither are the firing sequences restricted to the transitions left in the net. From this, we conclude that no confusion can be added.

Finally, consider the transition in the postset of $\bar{\mathbf{c}}$ in Figure 8.11. After the removal of $\bar{\mathbf{g}}$ and $\bar{\mathbf{f}}$ with $\textsc{prune}_{\preceq}$, this transition will have a single persistent place in its preset and a single persistent place in its postset. Therefore, this transition and the place in its postset can be removed by connecting $\bar{\mathbf{c}}$ directly to t_5. To do so, we define a fourth prune operation using *persistent chains*.

Definition 8.11 (Chain)**.** Let $N = (P, T, F)$ be a Petri net with persistent places. We call a *chain* a sequence $(x_n)_{1\leq n\leq k}$ with $k > 1$ such that $x_1, x_k \in P$ and

$$\forall 1 \leq i < k \;\; x_i^\bullet = \{x_{i+1}\} \wedge \{x_i\} = {}^\bullet x_{i+1}.$$

Definition 8.12 (Persistent chain)**.** A *persistent chain* is a chain such that all places in the chain are persistent. We denote with $\textsc{Ch}_N$ the set of maximal persistent chains in N.

An example of a persistent chain is shown in Figure 8.12a. Using the set of maximal persistent chains $\textsc{Ch}_N$, we define the prune operation

$$\textsc{prune}_{\textsc{Ch}}(N) \coloneqq (P', T', F'', m),$$

where

$$(P', T', F') = N \setminus \left\{x_n \;\middle|\; (x_n)_{1\leq n\leq k} \in \textsc{Ch}_N \wedge n > 1\right\},$$
$$F'' = F' \cup \left\{(x_1, y) \in (P' \times T') \;\middle|\; (x_n)_{1\leq n\leq k} \in \textsc{Ch}_N \wedge y \in x_k^\bullet\right\}.$$

This operation removes the chain, except for the first persistent place of the chain and connects it to the postset of the last place in the chain. The result of this operation on the persistent chain in Figure 8.12a is shown in Figure 8.12b.

Proposition 8.2. *A sequence of transitions σ in a Petri net N is a valid firing sequence if and only if σ with the transitions in the chains in $\textsc{Ch}_N$ removed is a valid firing sequence in $\textsc{prune}_{\textsc{Ch}}(N)$.*

Proof. Let N be a Petri net containing a maximal persistent chain $C = (x_i)_{1 \leq i \leq p} \in \textsc{Ch}_N$ with the set of transitions $\{s_1, \ldots, s_n\}$ and let $N' = (P', T', F'')$ be the net obtained by removing C from N as follows:

$$(P', T', F') = N \setminus \{x_i \mid i > 1\}$$
$$F'' = F' \cup \left\{(x_1, y) \;\middle|\; y \in x_p^\bullet\right\}$$

Additionally, let $\{p\} = {}^\bullet s_1$ and $\{q\} = s_n^\bullet$ and let p' be the place corresponding to p in N'. Then $(p')^\bullet = q^\bullet$.

First note that if a firing sequence σ does not contain all transitions $s_1, \ldots, s_n$, then firing $s_1, \ldots, s_n$ does not enable any transitions outside of the chain. Therefore, σ with the transitions in C removed is a valid firing sequence in N.

Now take a firing sequence of N containing all transitions $s_1, \ldots, s_n$:

$$m_0 \to^* m_1 \xrightarrow{s_1} m_2 \to^* m_3 \xrightarrow{s_3} m_4 \to^* \ldots \to^* m_n.$$

Since C is a chain, the transitions fired between s_1 and s_n must be concurrent to $s_1, \ldots, s_n$. Therefore, this firing sequence is equivalent to

$$m_0 \to^* m_1 \xrightarrow{s_1 \cdots s_n} m'_{n+1} \to^* m'_n,$$

where m'_1 is the first marking where s_1 is enabled. Call the transitions in this sequence:

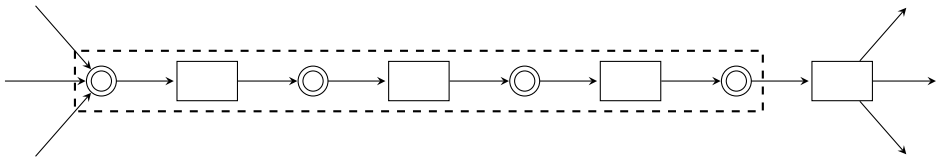
$$(t_1, \ldots, t_k, s_1, \ldots, s_n, t_{k+1}, \ldots, t_l).$$

We will prove that the sequence $(t_1, \ldots, t_k, t_{k+1}, \ldots, t_l)$ is a valid firing sequence in N'. Trivially, $(t_1, \ldots, t_k)$ is a valid firing sequence in N'. Firing t_k in N' will then mark p'. Since $(p')^\bullet = q^\bullet$ and therefore the transitions in $\{t_{k+1}, \ldots, t_l\}$ are enabled as after firing $(t_1, \ldots, t_k, s_1, \ldots, s_n)$ in N. Since the flow relation restricted to $t_{k+1}, \ldots, t_l$ is preserved, $(t_1, \ldots, t_k, s_1, \ldots, s_n, t_{k+1}, \ldots, t_l)$ is a valid firing sequence in N'.

Conversely, take any firing sequence $(t_1, \ldots, t_l)$ in N'. Let t_{k+1} be the first transition in the sequence such that $t_{k+1} \in (p')^\bullet$. If there is no such transition, then $(t_1, \ldots, t_l)$ is a firing sequence in N. We will now prove that if there is a $t_{k+1} \in (p')^\bullet$ then $(t_1, \ldots, t_k, s_1, \ldots, s_n, t_{k+1}, \ldots, t_l)$ is a valid firing sequence in N. It suffices to show that s_1 is enabled after firing t_k and does not introduce conflict and that t_{k+1} is enabled after firing s_n. In N', the firing sequence $(t_1, \ldots, t_k)$ enables t_{k+1}, therefore p' is marked after the firing sequence $(t_1, \ldots, t_k)$. Hence, firing $t_1, \ldots, t_k$ in N marks p. Since $\{p\} = {}^\bullet s_1$, the transition s_1 is therefore enabled. Furthermore, we have $\{s_1\} = p^\bullet$, therefore s_1 is not in direct conflict with other transitions. Firing $s_1, \ldots, s_n$ then marks q. Again, since $(p')^\bullet = q^\bullet$, the same transitions are enabled as after firing $t_1, \ldots, t_k$ in N'. Therefore, $(t_1, \ldots, t_k, s_1, \ldots, s_n, t_{k+1}, \ldots, t_l)$ is valid firing sequence in N.

We repeat the same argument to remove all chains in $\textsc{Ch}_N$. We then conclude by induction that a sequence of transitions σ in N is a valid firing sequence if and only if σ with the transitions in the chains in $\textsc{Ch}_N$ removed is a valid firing sequence in $\textsc{prune}_{\textsc{Ch}}(N)$. □

Note that the proposition above implies that direct conflict is preserved, since transitions a an b are in direct conflict if and only if for every firing sequence $(t_1, \ldots, t_n, a)$ there does not exist a firing sequence $(t_1, \ldots, t_n, a, b)$. Therefore, no confusion is introduced by the $\textsc{prune}_{\textsc{Ch}}(N)$ operation.



(a) A maximal persistent chain is represented in the dashed box.

(b) The same maximal persistent chain after applying $\textsc{prune}_{\textsc{Ch}}$.

Figure 8.12: A persistent chain and the result of pruning this chain.

With these four pruning operations in place, we define the combined pruning operation as follows:

$$\textsc{prune} \coloneqq \textsc{prune}_{\textsc{Ch}} \circ \textsc{prune}_{\preceq} \circ \textsc{prune}_{\lhd} \circ \textsc{prune}_{(\!|\cdot|\!)} \,.$$

Running example Applying this operation to the running example, we obtain the net $\textsc{prune}((\!|[\![N]\!]|\!))$ shown in Figure 8.13.

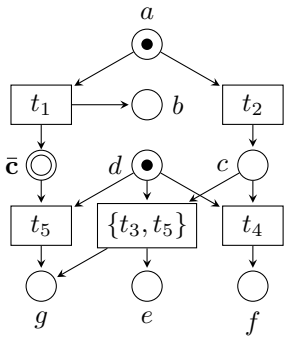


Figure 8.13: The net $\textsc{prune}((\!|[\![N]\!]|\!))$ for the running example.

8.6 Attaching Probabilities

With the s-cells method, a probabilistic net can be created directly without computing an event structure like with the branching cells method. Here, a method given in [5] is briefly explored.

Given a finite acyclic Petri net, a probability $\mathbb{P}_{\mathbb{C}}$ is attached to every transaction in an s-cell $\mathbb{C}$ such that

$$\sum_{\theta\,:\,\mathbb{C}} \mathbb{P}_{\mathbb{C}}(\theta) = 1.$$

Then these probabilities are applied to the net $[\![N]\!]$ by setting the probabilities for the transitions in T_{pos} equal to the probabilities of the corresponding transactions in N and setting the probabilities for the transitions in T_{neg} to 1. We then note that the the nets $(\!|[\![N]\!]|\!)$ and $[\![N]\!]$ have the same transitions, so the probabilities on $[\![N]\!]$ can directly be transferred to $(\!|[\![N]\!]|\!)$.

Bruni, Melgratti, and Montanari [5] have shown that it is possible to choose the probabilities $\mathbb{P}_{\mathbb{C}}$ such that they correspond to the probabilities in a Markov net.

Chapter 9

Conclusion

The s-cells method as defined in [5] is based on the assumption that the starting Petri net is acyclic. Even though the branching cells method relies on many of the same principles, it deals with cycles by only defining probabilities on the unfolding of the net.

The results of this paper are mostly practical, as they assist in the implementation of the s-cells method. First, the fact that s-cells can be computed as strongly connected components allows for the use of established algorithms. Second, the formalisation of the pruning operations allows for smaller nets in the output. A Python implementation of the s-cell method created using the findings in this paper can be found at `https://gitlab.com/tertsdiepraam/petrinet`.

Additionally, we have presented a definition of the removal of dependence operator $\ominus$ based on the dependence relation $\lhd$, which can be applied to locally finite nets, but coincides with the definition in [5] in the case of acyclic nets. This is a small first step in adapting the s-cells construction to locally finite nets. Future work could investigate how the definition of s-cells and the rest of the construction can be further adapted to locally finite nets.

Bibliography

[1] S. Abbes. "Probabilistic model for distributed and concurrent systems. Limit theorems and application to statistical parametric estimation". eng. PhD thesis. IRISA-Université de Rennes 1, Oct. 2004.

[2] S. Abbes and A. Benveniste. "Branching Cells as Local States for Event Structures and Nets: Probabilistic Applications". eng. In: *Foundations of Software Science and Computational Structures*. Ed. by V. Sassone. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 95–109. ISBN: 978-3-540-31982-5.

[3] S. Abbes and A. Benveniste. "True-concurrency probabilistic models: Markov nets and a law of large numbers". eng. In: *Theoretical Computer Science* 390.2 (2008), pp. 129–170. ISSN: 0304-3975.

[4] F. Bause and P. Kritzinger. *Stochastic Petri Nets – An Introduction to the Theory*. eng. Nov. 2013. ISBN: 3-528-15535-3.

[5] R. Bruni, H. Melgratti, and U. Montanari. "Concurrency and Probability: Removing Confusion, Compositionally". eng. In: *Logical Methods in Computer Science* 15 (4 Oct. 12, 2017). arXiv: 1710.04570.

[6] A. Einstein. "Zur Elektrodynamik bewegter Körper". In: *Annalen der Physik* 322.10 (1905), pp. 891–921. DOI: 10.1002/andp.19053221004.

[7] L. Lamport. "Time, Clocks, and the Ordering of Events in a Distributed System". In: *Commun. ACM* 21.7 (July 1978), pp. 558–565. ISSN: 0001-0782. DOI: 10.1145/359545.359563.

[8] M. Nielsen, G. Plotkin, and G. Winskel. "Petri nets, event structures and domains, part I". eng. In: *Theoretical Computer Science* 13.1 (1981), pp. 85–108. ISSN: 0304-3975.

[9] C. A. Petri. "Kommunikation mit Automaten". ger. PhD thesis. Universität Hamburg, 1962.

[10] V. Sassone, M. Nielsen, and G. Winskel. "Models for concurrency: towards a classification". In: *Theoretical Computer Science* 170.1 (1996), pp. 297–348. ISSN: 0304-3975. DOI: 10.1016/S0304-3975(96)80710-9.

[11] R. Tarjan. "Depth-First Search and Linear Graph Algorithms". In: *SIAM Journal on Computing* 1.2 (1972), pp. 146–160. DOI: 10.1137/0201010.

[12] D. Varacca, H. Völzer, and G. Winskel. "Probabilistic event structures and domains". eng. In: *Theoretical Computer Science* 358.2 (2006). Concurrency Theory (CONCUR 2004), pp. 173–199. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2006.01.015.

[13] G. Winskel. "A new definition of morphism on Petri nets". eng. In: *STACS 84*. Ed. by M. Fontet and K. Mehlhorn. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984, pp. 140–150. ISBN: 978-3-540-38805-0.

Notation

Numbers

$\mathbb{N}$	natural numbers
$\mathbb{N}_\infty$	natural numbers including infinity

Logical

$\circ$	function composition
$:=$	definition
$\exists$	existential quantification
$\forall$	universal quantification
Id	identity relation
$\iff$	bi-implication
$\implies$	implication
R^{-1}	inverse of the relation R
$\restriction$	domain restriction
$\exists!$	unique existential quantification
$\vee$	or
$\wedge$	and
R^*	transitive and reflexive closure
R^+	transitive and irreflexive closure

Sets

$\setminus$	set difference
$\emptyset$	empty set

$\in$	membership
$\cap$	intersection
$\subset$ / $\subseteq$	strict subset / subset
$\times$	Cartesian product
$\cup$	union

Petri nets

$\bar{p}$	negative place, page 42
$\#$	conflict relation, page 14
$\#_0$	direct conflict relation, page 14
$\triangleleft$	dependence, page 39
$\setminus$	removal of nodes, page 12
$\mathcal{D}^\circ$	final places, page 2
${}^\circ\mathcal{D}$	initial places, page 2
$\mathbf{P}$	set of persistent places, page 34
$\mathbf{p}$	persistent place, page 34
$p^\bullet$	postset, page 11
${}^\bullet p$	preset, page 11
$\preceq$	causality relation, page 12
$\overline{P}$	set of negative places, page 42

Event structures

$[e]$	smallest configuration containing e, page 20
$[e[$	smallest configuration enabling e, page 20
$\#$	conflict relation, page 19
$\mathcal{E}^v$	future of the configuration v, page 20
$\preccurlyeq$	causality relation, page 19
$\#_\mu$	immediate conflict relation, page 20
$\mathcal{V}_\mathcal{E}$	finite configurations of $\mathcal{E}$, page 20
$\Omega_\mathcal{E}$	maximal configurations of $\mathcal{E}$, page 20