A Unified Semantic Equivalence for Real-Time and Liveness Properties in Concurrent Systems

Alexander Eyre

Master of Science
School of Informatics
University of Edinburgh

2025

Abstract

In the formal analysis of concurrent systems, semantic equivalences are crucial tools for determining when two systems exhibit the same essential behaviour. Whilst the literature is rich with such equivalences, including those that preserve real-time properties and those that preserve liveness properties, there exists no relation that addresses both simultaneously. This paper bridges this gap by introducing the **Enabling Preserving ST-bisimulation** (EPST-bisimulation), a novel semantic equivalence for unsafe Petri nets. EPST-bisimulation unifies the principles of ST-bisimulation and Ep-bisimulation by defining equivalence over states that track both in-progress transition sequences and the concurrency structure of enabled transitions.

We rigorously establish the theoretical properties of EPST-bisimulation, proving it is an equivalence relation and, crucially, the coarsest equivalence finer than both ST- and Ep-bisimulation, cementing its natural place in the semantic hierarchy. Furthermore, we demonstrate its suitability for compositional reasoning by proving that EPST-bisimulation is a congruence for the Algebra of Communicating Processes (ACP), and is preserved under action refinement.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Alexander Eyre*)

Acknowledgements

I would like to thank my supervisor, Rob van Glabbeek, for his patience and assistance in navigating these ideas.

Table of Contents

Chapter 1

Introduction

A fundamental question in the analysis of concurrent systems is that of *equivalence*, the question of when we consider two systems to be “the same” in some particular sense. Semantic equivalences provide a formal basis for answering this question. There exists a wide spectrum of equivalences, ranging from *trace equivalences* which only compare possible sequences of actions, to more discriminating *branching-time equivalences* such as bisimulations [3]. Bisimulations, originally proposed by David Park [9], approach equivalence from the perspective that processes are equivalent if they can simulate each other’s behaviour. However, traditional strong bisimulation is often insufficient and fails to distinguish between systems that still exhibit critical behavioural differences. We treat two types of critical behaviour in this paper: real-time behaviour and liveness.

Firstly, standard models often assume that actions are instantaneous. In reality, actions take time to execute, network requests may have to idle, cache misses may require spinning up slow storage, even simple arithmetic has some execution time. *ST-bisimulation* [11] addresses this by extending the notion of a system’s state to include not just the current place configuration (a *marking*), but also the ordered sequence of transitions that have started but not yet finished. This allows it to distinguish between processes with different real-time characteristics, for example different minimal execution times.

Secondly, we are often interested in the *liveness properties* of a system, that is guarantees that a system will eventually exhibit some good behaviour, such as that once a network request has been received, the system eventually replies. This requires some level of assumption on the progress of the system and its components, as without any assumptions at all there is no guarantee that even the most trivial system ever moves from its initial state. *Justness* is a particularly useful such assumption, weaker than the assumption of *fairness*. The former requires that once a transition is persistently enabled, it will either eventually be executed or interfered with, whereas the latter states that any persistently enabled action will eventually be executed [4]. To preserve this property, *Ep-bisimulation* was introduced [2]. This bisimulation augments the equivalence relation with a relation on the enabled transitions of the two systems in each state, ensuring that their concurrency structures are equivalent.

Whilst both ST-bisimulation and Ep-bisimulation are powerful, they address orthogonal concerns. A comprehensive analysis of complex systems requires consideration of real-time behaviour and liveness properties simultaneously. In this paper we bridge this gap by introducing a new semantic equivalence: the *Enabling Preserving ST-bisimulation*, or *EPST-bisimulation*. This novel equivalence unifies the principles of its predecessors by relating states that consist of both an ST-state (a marking and a sequence of firing transitions) and a concurrency-preserving relation matching the enabled transitions.

This work formally defines EPST-bisimulation for unsafe Petri nets and rigorously establishes its properties. We demonstrate that it is an equivalence relation strictly finer than both ST- and Ep-bisimulation. Crucially, we prove that it is the *coarsest* such equivalence relation to satisfy these conditions, cementing its position in the lattice of semantic equivalences. Furthermore, we show that EPST-bisimulation is a *congruence* for the operators of the Algebra of Communicating Processes (ACP), a formal algebra for the specification of concurrent processes [1], and that it is preserved under *action refinement*, a more complex notion that involves the substitution of single transitions with more complex processes. With these properties in place we propose that EPST-bisimulation is an appropriate relation for compositional reasoning and iterative hierarchical system design and verification.

This paper is structured as follows: Chapter 2 provides the necessary background on Petri nets and ACP; Chapter 3 reviews the foundational concepts of ST- and Ep-bisimulations, adapting them for the context of unsafe nets; Chapter 4 introduces the central contribution, the EPST-bisimulation, and proves its fundamental properties; Chapters 5 and 6 establish its congruence for ACP and its preservation under action refinement respectively; Chapter 7 provides illustrative examples; and Chapter 8 concludes with a summary of contributions and directions for future research.

Chapter 2

Preliminaries

2.1 Petri Nets

Petri nets are a model of concurrency first proposed by Carl Adam Petri in order to model chemical processes, and subsequently utilised in his thesis [10] to model concurrent processes. They are directed bipartite graphs made up of places and transitions, with directed edges from places to transitions and from transitions to places. Tokens, markers representing computation, exist inside places and flow along the directed edges through transitions to indicate execution. Transitions execute by consuming one token from each one of their input places and finish by placing a token into each of their output places.

Formally, over a set of actions A, we define a Petri net N as a tuple

$$N = (S, T, F, M^{\text{in}}, \ell),$$

where S is the set of places or states, T is the set of transitions, F is the flow relation, M^{in} is the initial marking or the initial state of the system before any tokens have moved, and $\ell : T \to A$ is the labelling function that determines the action output when a given transition fires.

The flow relation $F \subseteq (S \times T) \cup (T \times S)$ contains pairs of places and transitions and indicates the directed edge structure of the net. For example, $(s,t) \in F$ implies that there is an edge from place s to transition t, such that s is in the pre-set of t, and $(t,s) \in F$ indicates that s is in the post-set of t.

The markings of a net $N = (S, T, F, M^{\text{in}}, \ell)$, often denoted M, are configurations of the net and specify the positions of tokens within the places of the net. Petri nets are broadly split into two main categories, known as safe and unsafe nets. **Safe nets** restrict markings to allow only one token per place, and typically utilise a straightforward subset of the places to represent a marking, i.e. $M \subseteq S$. **Unsafe nets** on the other hand have no restriction on the number of tokens that can occupy a single place and utilise more complex structures such as multisets to represent markings. Markings in this case are typically a structure such as multisets, denoted $M : S \to \mathbb{N}$ or $M \in \mathbb{N}^S$, where for a given place $s \in S$, $M(s)$ returns the number of tokens in place s in the marking

M. Although an abuse of notation, we often refer to both the empty safe and unsafe markings with the same symbol $\varnothing$ throughout this paper. For a safe net $\varnothing$ refers to the set containing no elements with a cardinality of zero, and for unsafe nets $\varnothing$ refers to the function $M : S \to \mathbb{N}$ that returns 0 for every $s \in S$.

For a given transition t in a net $N = (S,T,F,M^{\text{in}},\ell)$ such that $t \in T$, the pre- and post-sets of t refer to the places it interacts with. We refer to the pre-set of t as ${}^*t \subseteq S$ to mean the set of places from which t must consume a token in order to fire, and the post-set of t as $t^* \subseteq S$ to mean the set of places t inserts a token into once it has finished firing.

Definition 2.1. For a transition t in a Petri net $N = (S,T,F,M^{\text{in}},\ell)$, we define the pre-set ${}^*t$, and the post-set t^* as follows,

$$\begin{aligned} {}^*t &= \{s \in S \mid (s,t) \in F\} \subseteq S \\ t^* &= \{s \in S \mid (t,s) \in F\} \subseteq S. \end{aligned}$$

When talking about transitions affecting the markings of a net $N = (S,T,F,M^{\text{in}},\ell)$, we utilise the notation $|\rangle$. For example, if we have a marking M of N and a transition t enabled in M, the firing of which results in some new marking M', we would write $M|t\rangle M'$. We are often only concerned with whether a transition t is enabled in a marking M, and in this case we say that t is in the set of enabled transitions of M, $t \in \text{en}(M)$, or simply $M|t\rangle$.

2.1.1 Labelled Transition Systems

Petri nets offer a model of concurrency that supports *true concurrency*, a key advantage over structures such as Labelled Transition Systems (LTSs), which we also considered. An LTS represents a system as a state machine where actions label the transitions between states, modelling concurrency through the interleaving of actions [5]. Whilst any Petri net has a corresponding LTS (known as its reachability graph [6]), the net representation retains explicit information about causal dependencies and concurrency. Examples exist of methods to augment LTSs with causality information, such as LTSs with Successors (LTSSs) in Glabbeek, Höfner, and Wang [2]. LTSSs include concurrency information via a successor relation $\leadsto \subseteq T \times T \times T$, where $t \leadsto_v t'$ indicates that t and v are concurrent and that t' is a variant of t enabled after v fires. We opted for a model of concurrency that natively supports such information. For example, two transitions being concurrently enabled is directly observable in a net, whereas in an LTS this is represented by offering two different interleavings of those actions. This ability to model true concurrency rather than reducing it to interleaving is crucial for defining and preserving properties such as justness, central to Ep- and EPST-bisimulation.

2.2 Algebra of Communicating Processes

The Algebra of Communicating Processes (ACP) is an algebraic language for specifying and reasoning about concurrent systems developed by Jan Bergstra and Jan Willem

Klop [1]. The syntax of ACP is as follows, defined over some set of visible actions A in addition to the internal action symbol τ:

1. ϵ successful termination;
2. δ deadlock;
3. a the action constant for each action $a \in A$;
4. $P.Q$ sequential composition, i.e. do P and then after P successfully terminates, move on to Q;
5. $P + Q$ choice, either do P or Q but not both;
6. $P||Q$ parallel composition, do P and Q in parallel, sometimes denoted $P||_\gamma Q$ to indicate that γ is the communication function for this parallelisation;
7. $\partial_H(P)$ restriction or encapsulation for each set of visible actions $H \subseteq A$, stop the process P performing any action in H;
8. $\tau_I(P)$ abstraction for each visible action in the set $I \subseteq A$, turn every action $a \in I$ into the internal action τ.

For each application, a partial communication function $\gamma : A \times A \rightarrow A$ is chosen that specifies for two visible actions $a, b \in A$ whether they synchronise. If $\gamma(a, b)$ is defined it allows the processes to synchronise on a and b, and determines what the result of their synchronisation should be. The internal action τ cannot take part in synchronisation.

These operators can also be defined in terms of Petri nets, and throughout this paper we utilise the Petri net semantics of ACP proposed by Van Glabbeek and Vaandrager [11].

Chapter 3

Foundational Semantic Equivalences

3.1 ST-Bisimulation

3.1.1 Real-time behaviour

When modelling real-world systems, one of the first enhancements to move the model closer to their real-world counterparts is to drop the assumption that actions are instantaneous. We accomplish this by allowing for some processing time between the beginning and end of each action. However, this additional complexity poses a challenge for standard bisimulations. For example, consider the two processes represented by two nets that are strongly bisimilar in Figure 3.1.

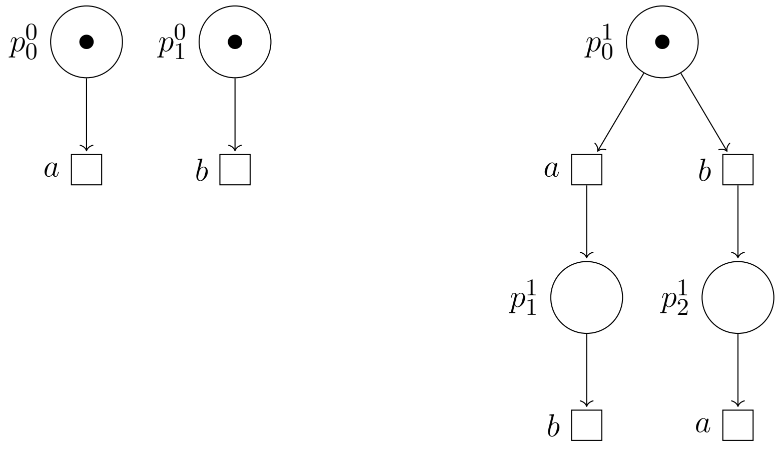


Figure 3.1: nets

Intuitively, the initial markings are related, and the markings after the firing of a or b are related. Formally, we can define a strong bisimilarity $\mathscr{R}$ as follows,

$$\mathscr{R} := \{(\{p_0^0, p_1^0\}, \{p_0^1\}), (\{p_0^0\}, \{p_2^1\}), (\{p_1^0\}, \{p_1^1\}), (\varnothing, \varnothing)\}.$$

Remark. Although we are considering unsafe nets throughout, for the purposes of clarity we will use safe-marking set notation in this example.

However, consider what happens when we allow actions to take some execution time to complete, these nets exhibit different *real-time behaviour*. For example, they have different minimal execution times. If a takes one time unit to complete, and b two, then P_0 has a minimal execution time of two as a and b can fire simultaneously, whereas P_1 can only execute one action after the other has completed, and has a minimum execution time of three.

Although we are allowing actions to take some time to execute, it is not actually relevant to our modelling from a bisimulation perspective how long each action takes, only that it can take some arbitrary amount of time. The concept of an **ST-state** seeks to address this by augmenting the typical marking description for the state of a net with an ordered firing sequence that contains the actions currently in-progress, which can finish at any arbitrary point.

Definition 3.1 (ST-state)**.** In the context of nets with non-instantaneous actions, an ST-state of a Petri net $N = (S, T, F, M^{\text{in}}, \ell)$ is a tuple consisting of a marking of that net and a sequence of transitions that are currently in-progress in the order they started firing.

$$[M, \sigma] \in \mathbb{N}^S \times T^*.$$

3.1.2 Paths and ST-Paths

When reasoning about the whole execution of a net we often want to consider a whole sequence of events that occur to bring the net from one state to another, with the specific intermediary states it went through to do so, referred to as a *path*.

Definition 3.2 (Paths for Petri nets)**.** A *path* in a Petri net is an alternating sequence $M_0 u_0 M_1 u_1 M_2 \ldots$ of markings and transitions, starting with a state and either being infinite or ending with a state, such that $M_i[u_i\rangle M_{i+1}$ for all relevant i. The *length* $l(\pi) \in \mathbb{N} \cup \{\infty\}$ of a path π is the number of transitions in it. If π is a path, $\hat{\pi}$ is the sequence of transitions in it.

However, we encounter additional complexity when we consider nets with ST-states rather than simple markings. Primarily two things: we need some way to indicate that a transition has begun and finished firing separately; and in the case of unsafe nets, we need a way to indicate *which* transition has finished firing, as there is no guarantee that execution is FIFO. A transition occurring later may finish firing first and still form a valid path. To address these issues, we introduce the concept of an **ST-path**.

Definition 3.3 (ST-path)**.** An **ST-path** is an alternating sequence of ST-states and intermediary transitions indicating the firing and finishing of transitions. For each transition $t \in T$, t^+ indicates the start of firing for that transition, and t^{-k} indicates that the k^{th} t transition has finished firing, where k is 1-indexed and counted from left to right. For example, if we have some firing sequence (t, a, b, t), a t^{-1} intermediary transition indicates the first t transition has finished, whereas t^{-2} indicates the same for the second. When left unqualified, t^- is equivalent to t^{-1}.

Formally, an ST-path π is an alternating sequence of ST-states and intermediary transitions as described above, starting with an ST-state and being either infinite or ending with an ST-state,

$$\pi = [M_0,\sigma_0]\alpha_0[M_1,\sigma_1]\alpha_1[M_2,\sigma_2]\dots[M_i,\sigma_i]\alpha_i[M_{i+1},\sigma_{i+1}]\dots$$

Where the length of the path $l(\pi) \in \mathbb{N} \cup \{\infty\}$ is the number of intermediary transitions in the path. One of the following holds for all $i < l(\pi)$:

1. **Transition Starting** $\alpha_i = t^+$: A transition $t \in T$ that is enabled in the marking M_i begins to fire, and therefore $M_{i+1} = M_i - {}^*t$ and $\sigma_{i+1} = \sigma_i * t$.
2. **Transition Ending** $\alpha_i = t^{-k}$: A transition $t \in T$ that is currently firing, that is $\sigma_i = \rho * t * \eta$ with t being the k^{th} t transition in σ_i as described previously. In this case, $M_{i+1} = M_i + t^*$ and $\sigma_{i+1} = \rho * \eta$.

3.1.3 Defining ST-Bisimulation

With the concept of an ST-state in mind, we can introduce the **ST-bisimulation**. Originally described by Van Glabbeek and Vaandrager [11], it strengthens the notion of traditional bisimulation to utilise ST-state information to ensure that real-time behaviour is preserved between processes.

Definition 3.4 (ST-bisimulation on safe nets [11])**.** Let two nets be given as

$$N_0 = (S_0,T_0,F_0,M_0^{\text{in}},\ell_0) \quad \text{and} \quad N_1 = (S_1,T_1,F_1,M_1^{\text{in}},\ell_1).$$

A relation

$$R \subseteq (\mathscr{P}(S_0)\times T_0^*)\times(\mathscr{P}(S_1)\times T_1^*)$$

is an **ST-bisimulation** if the following conditions hold:

1. $(M_0^{\text{in}},\epsilon)R(M_1^{\text{in}},\epsilon)$;
2. If $(M_0,\sigma_0)R(M_1,\sigma_1)$ and $M_0[t_0\rangle$ then there is a $t_1 \in T_1$ such that $\ell_0(t_0) = \ell_1(t_1), M_1[t_1\rangle$ and $(M_0 - {}^*t_0, \sigma_0 * t_0)R(M_1 - {}^*t_1, \sigma_1 * t_1)$;
3. the same as 2 with the roles of N_0 and N_1 reversed;
4. If $(M_0,\sigma_0 * t_0 * \rho_0)R(M_1,\sigma_1 * t_1 * \rho_1)$ and $|\sigma_0| = |\sigma_1|$ then $(M_0 + t_0^*, \sigma_0 * \rho_0)R(M_1 + t_1^*, \sigma_1 * \rho_1)$;
5. the same as 4 with the roles of N_0 and N_1 reversed;
6. $(M_0,\sigma_0)R(M_1,\sigma_1) \implies ((M_0,\sigma_0) = (\varnothing,\epsilon) \iff (M_1,\sigma_1) = (\varnothing,\epsilon))$.

As we are working with unsafe nets throughout this paper, this definition requires lifting to unsafe nets, we present the following modification of the ST-bisimulation and claim it exhibits the same goodness properties as its safe counterpart.

Definition 3.5 (ST-bisimulation)**.** Let two unsafe nets be given as

$$N_0 = (S_0,T_0,F_0,M_0^{\text{in}},\ell_0) \quad \text{and} \quad N_1 = (S_1,T_1,F_1,M_1^{\text{in}},\ell_1).$$

A relation

$$R \subseteq (\mathbb{N}^{S_0}\times T_0^*)\times(\mathbb{N}^{S_1}\times T_1^*)$$

is an **ST-bisimulation** if the following conditions hold:

1. $(M_0^{\text{in}}, \epsilon) R (M_1^{\text{in}}, \epsilon)$;
2. If $(M_0, \sigma_0) R (M_1, \sigma_1)$ and $M_0[t_0\rangle$ then there is a $t_1 \in T_1$ such that $\ell_0(t_0) = \ell_1(t_1), M_1[t_1\rangle$ and $(M_0 - {}^*t_0, \sigma_0 * t_0) R (M_1 - {}^*t_1, \sigma_1 * t_1)$;
3. the same as 2 with the roles of N_0 and N_1 reversed;
4. If $(M_0, \sigma_0 * t_0 * \rho_0) R (M_1, \sigma_1 * t_1 * \rho_1)$ and $|\sigma_0| = |\sigma_1|$ then $(M_0 + t_0^*, \sigma_0 * \rho_0) R (M_1 + t_1^*, \sigma_1 * \rho_1)$;
5. the same as 4 with the roles of N_0 and N_1 reversed;
6. $(M_0, \sigma_0) R (M_1, \sigma_1) \implies ((M_0, \sigma_0) = (\varnothing, \epsilon) \iff (M_1, \sigma_1) = (\varnothing, \epsilon))$.

Remark. This definition extends the original for safe nets to the unsafe case by representing markings as multisets. The fundamental properties of ST-Bisimulation, such as real-time consistency, are preserved under this extension as the core mechanics of tracking firing sequences remain unchanged.

3.2 Ep-Bisimulation

3.2.1 Progress, Fairness, and Justness

When reasoning over concurrent systems we often want to make statements on the good behaviour they eventually exhibit, such as the eventual execution of some action. We call this type of property a *liveness property*.

For example, consider the following program:

```
x = 0;
while(true) {
  x = x + 1;
}
```

A liveness property of this program is that eventually $x \geq n$ for any number $n \in \mathbb{N}$. Intuitively this property is true under the typical set of implicit assumptions, primarily in this case *progress*. Progress is the assumption that if a process is in a state in which it can perform an action it will eventually perform an action. Applying this to the previous example, progress is the assumption that the program will never stall at any arbitrary `x` assignment as there is always another available to the system.

But if we want to reason about processes with some concurrency then progress is an insufficient assumption, for example if we have the previous example running in parallel with some trivial assignment as follows:

```
x = 0;                        ||
while(true) {                 ||   y = 1;
  x = x + 1;                  ||
}
```

If we view this whole process only through the lens of progress, then the infinite execution path containing only `x = x + 1` actions is a valid path, as in every state in

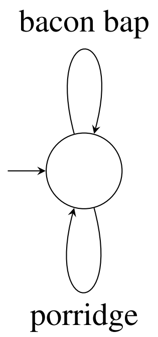


Figure 3.2: LTS representation of the breakfast choice.

which the process admits a visible action it performs one.

How we view the validity of this path depends on the model of concurrency we employ. If we opt for *interleaving semantics*, that is viewing parallel execution as a single thread that at each step of execution chooses arbitrarily between its available actions, such that $a||b$ is equivalent to $a.b+b.a$, then one can argue that the previous path is possible but unlikely.

However, this is not how modern computation occurs at all. If we define a process $a||b$ such that a and b do not interfere with each other, then they can both happen at the same time on multiple threads. This school of thought is known as *true concurrency*. But even if we work under the assumption of true concurrency, the process still satisfies progress as it never arbitrarily halts whilst actions are still available to it—we need a different sort of assumption.

Enter *fairness*, informally this is that if a process has a choice between two actions, say a and some other action and we encounter this choice infinitely often, then the process will eventually choose a.

Definition 3.6. A process is *fair* if and only if every transition enabled infinitely often is eventually taken [4].

If we now assume fairness, then the parallel process discussed above cannot take an action path perpetually excluding `y = 1` as it is enabled infinitely often, and the process will therefore eventually execute it.

However, fairness is often an overly strong assumption that does not match with real-world systems. A classic example from Glabbeek, Höfner, and Wang [2] illustrates this property in an intuitive, familiar setting: breakfast. Imagine a person called Alice choosing between two classic Scottish breakfast options each morning: a bacon bap or a bowl of porridge. We could specify this process with the guarded recursive specification

$$P = (\text{bacon bap} + \text{porridge}).P$$

equivalent to the LTS in Figure 3.2.

If we assume fairness, then the statement that at some point Alice will have had both a bacon bap and porridge for breakfast is true, she is offered each item infinitely often (globally in fact), but in the real-world this is not necessarily the case. For example, Alice could be a vegetarian and therefore it is a perfectly valid statement to say that she will opt for porridge every single time she is offered a choice.

Enter *justness*, a weaker fairness-style assumption that seeks to remedy this. Justness states that

> *Once a non-blocking transition is enabled that stems from a set of parallel components, one (or more) of these components will eventually partake in a transition.* [4]

Our breakfast example contains only one parallel component, and therefore if we assume justness rather than fairness it is a valid property of the system that Alice can always opt for porridge and maintain her dietary preferences.

Now let us return to our previous, more computational example with the infinite addition and single assignment. If we make the reasonable assumption that each x assignment does not interfere with, or is concurrent with the y assignment, then the previous path ignoring the y component is not *just* as it involves a parallel component never partaking in an action whilst one is enabled. Therefore, if we make a *justness assumption*, then no such paths are possible and the liveness property that eventually $y = 1$ is true.

In order to formalise this notion that two transitions are concurrent we introduce the notion of the *asymmetric concurrency operator* $\smile\!\!\!\bullet$. If we have two transitions a and b, we say $a \smile\!\!\!\bullet\, b$ to mean that a is unaffected by b, where by unaffected we mean that if b fires it does not affect the ability of a to fire.

Remark. There also exists the obvious symmetric extension of this in the form of $\smile$, where $a \smile b \iff a \smile\!\!\!\bullet\, b \wedge b \smile\!\!\!\bullet\, a$, but in the context of Petri nets these two operators are equivalent.

Definition 3.7 (Asymmetric Concurrency Operator)**.** Given a Petri net $N = (S,T,F,M^{\text{in}},\ell)$ and two transitions $t,u \in T$ enabled in some marking M, we say that $t \smile\!\!\!\bullet\, u$, read as t is concurrent with u if and only if:

$$({}^*t + {}^*u) \leq M.$$

Or that the tokens in marking M are sufficient to allow both t and u to fire without taking into account the tokens output by either transition.

Remark. There is a competing definition of concurrency that defines transitions as concurrent if they do not share any of the same places in their pre-set. That is for a net $N = (S,T,F,M^{\text{in}},\ell)$ and transitions $t,u \in T$, we say $t \smile\!\!\!\bullet\, u$ if and only if:

$${}^*t \cap {}^*u = \varnothing.$$

On safe nets this definition and Definition 3.7 are identical, but diverge on unsafe nets. We opt for the definition given previously as we feel it better encapsulates the notion of concurrency with places containing multiple tokens, although in most applications the two definitions function identically. The reader may choose to adopt whichever definition they find most intuitive unless otherwise stated.

3.2.2 Defining Ep-bisimulation

We are now in a position to introduce the *enabling-preserving bisimulation* (Ep-bisimulation). Originally proposed by Glabbeek, Höfner, and Wang [2] on a structure known as Labelled Transition Systems with Successors (LTSSs), this equivalence allows for reasoning over liveness properties by preserving justness.

The core idea of the Ep-bisimulation is to augment each related marking with an additional transition relation R that relates all enabled transitions in both nets in the related markings. The crucial condition is that this relation R must itself preserve the concurrency between all related transitions, ensuring the concurrency structures of the two processes is isomorphic. This property must then be maintained after a transition fires. That is, if we have two transitions t_0 and t_1 in the first and second nets respectively that are related by the transition relation R, written t_0Rt_1, then the states resulting from t_0 and t_1 firing must also be related for some relation R'. This new transition relation R' must then preserve the concurrency structure from R, that is if we have some other transitions t and u in the first and second nets respectively such that tRu, and $t \smile\!\!\bullet\, t_0$ or $u \smile\!\!\bullet\, t_1$, then it must be that $tR'u$.

This enforces that the two systems have an equivalent concurrency structure, ensuring that if one system is *just*, then the other must be also, ergo that they share the same set of liveness properties.

Definition 3.8 (Ep-bisimulation on unsafe nets)**.** Let two unsafe nets be given as

$$N_0 = (S_0, T_0, F_0, M_0^{\text{in}}, \ell_0) \quad \text{and} \quad N_1 = (S_1, T_1, F_1, M_1^{\text{in}}, \ell_1).$$

A relation $\mathscr{R} \subseteq \mathbb{N}^{S_0} \times \mathbb{N}^{S_1} \times \mathscr{P}(T_0 \times T_1)$ is an **Ep-bisimulation** if the following hold:

1. If $(M_0, M_1, R) \in \mathscr{R}$, then for the set of enabled transitions $\text{en}(M)$,

   $$(\forall t \in \text{en}(M_0) \exists u \in \text{en}(M_1).\, tRu) \wedge (\forall u \in \text{en}(M_1) \exists t \in \text{en}(M_0).\, tRu)$$

   and for all transitions $t, u \in T_0$ and $v, w \in T_1$:

   $$tRu \wedge vRw \implies (t \smile\!\!\bullet\, v \iff u \smile\!\!\bullet\, w).$$

2. There exists a relation R such that $(M_0^{\text{in}}, M_1^{\text{in}}, R) \in \mathscr{R}$.
3. If $(M_0, M_1, R) \in \mathscr{R}$, and t_0 and t_1 are enabled transitions in N_0 and N_1 respectively such that t_0Rt_1, then:
   (a) $\ell_0(t_0) = \ell_1(t_1)$.
   (b) There exists a relation R' such that $(M_0 - {}^*t_0 + t_0^*,\ M_1 - {}^*t_1 + t_1^*,\ R') \in \mathscr{R}$ that preserves concurrency:
      (i) If tRu and $t \smile\!\!\bullet\, t_0$, then $tR'u$.
      (ii) If tRu and $u \smile\!\!\bullet\, t_1$, then $tR'u$.
4. Same as above, with the roles of N_0 and N_1 reversed.

Chapter 4

EPST-Bisimulation

In the previous chapter, we explored two independent enhancements to the traditional strong bisimulation. ST-bisimulation provides a framework for reasoning about real-time behaviour by considering non-instantaneous duration of actions. In parallel, Ep-bisimulation offers a way to preserve justness by tracking the concurrency structure of enabled transitions. However, we are often interested in both properties, and to properly model and reason about these systems we need a single, unified equivalence that combines the strengths of both approaches. To this end, in this chapter we introduce the core contribution of this paper, the Enabling Preserving ST-bisimulation or **EPST-bisimulation**.

This chapter is structured as follows: we start by formally defining EPST-bisimulation and establish a number of basic properties, such as the fact it forms an equivalence relation and is finer than the two bisimulations we seek to combine; we then move on to more complex properties such as the fact it is the *coarsest* equivalence finer than both Ep- and ST-bisimulation; and finally we prove that it preserves justness and is real-time consistent. This serves as a solid theoretical foundation on which we can base the more complex proofs that appear in Chapters 5 and 6.

4.1 Defining EPST-Bisimulation for Unsafe Nets

EPST-bisimulation operates on triples $([M_0,\sigma_0],[M_1,\sigma_1],R)$ that combine the tracking of in-progress transitions with a concurrency-preserving transition relation.

- The **ST-state pair** $([M_0,\sigma_0],[M_1,\sigma_1])$ preserves real-time consistency by tracking the system's marking and the sequence of currently firing actions.
- The **transition relation** R preserves liveness properties by providing a matching between the enabled transitions of both systems that strictly preserves their concurrency structure. This ensures that any two transitions in one system are concurrent if and only if their counterparts in the other system are also concurrent.

The rules of the bisimulation maintain these components cohesively. Starting an action requires a matched move and a new concurrency-preserving relation R' on the remaining transitions. Finishing an action updates the ST-state accordingly. This dual-component state allows EPST-bisimulation to make finer distinctions than either ST- or Ep-bisimulation could alone.

Definition 4.1 (EPST-bisimulation)**.** Given two nets N_0 and N_1, a relation

$$\mathscr{R} \subseteq (\mathbb{N}^{S_0} \times T_0^*) \times (\mathbb{N}^{S_1} \times T_1^*) \times \mathscr{P}(T_0 \times T_1)$$

is an **EPST-bisimulation** if the following conditions hold:

1. There exists a relation R such that $([M_0^{\text{in}}, \epsilon], [M_1^{\text{in}}, \epsilon], R) \in \mathscr{R}$.
2. If $([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R}$, the relation R must provide a total matching on all enabled transitions and preserve the concurrency structure of the two processes. Formally we can express that R is a total matching as

   $$(\forall t_0 \in \text{en}(M_0) \exists t_1 \in \text{en}(M_1).t_0 R t_1) \wedge (\forall t_1 \in \text{en}(M_1) \exists t_0 \in \text{en}(M_0).t_0 R t_1),$$

   and that R preserves the concurrency structure such that for transitions $t, v \in T_0$ and $u, w \in T_1$

   $$tRu \wedge vRw \implies (t \smile\!\!\!\bullet\, v \iff u \smile\!\!\!\bullet\, w)\,.$$

3. If $([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R}$ and $t_0 \in \text{en}(M_0)$, $t_1 \in \text{en}(M_1)$ such that $t_0 R t_1$, then it must be that:
   (a) The labels match: $\ell_0(t_0) = \ell_1(t_1)$.
   (b) There exists a new relation R' such that the resulting ST-state is in the bisimulation:

   $$([M_0 - {}^*t_0, \sigma_0 * t_0], [M_1 - {}^*t_1, \sigma_1 * t_1], R') \in \mathscr{R}.$$

   (c) The new relation R' preserves the old relation for concurrent transitions:
      (i) If tRu and $t \smile\!\!\!\bullet\, t_0$, then $tR'u$.
      (ii) If tRu and $u \smile\!\!\!\bullet\, t_1$, then $tR'u$.
4. The symmetric version of Clause 3 must also hold (i.e., for any move by N_1, N_0 must match).
5. If $([M_0, \sigma_0 * t_0 * \rho_0], [M_1, \sigma_1 * t_1 * \rho_1], R) \in \mathscr{R}$ with $|\sigma_0| = |\sigma_1|$, then there exists some relation R' such that:

   $$([M_0 + t_0^*, \sigma_0 * \rho_0], [M_1 + t_1^*, \sigma_1 * \rho_1], R') \in \mathscr{R}.$$

6. If $([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R}$, then the two states must agree on termination:

   $$(M_0, \sigma_0) = (\varnothing, \epsilon) \iff (M_1, \sigma_1) = (\varnothing, \epsilon).$$

4.2 Fundamental Properties

Starting from the most basic properties, let us convince ourselves that the relational form of EPST-bisimulation is an *equivalence relation*.

Theorem 4.1. $\underline{\leftrightarrow}_{EPST}$ is an equivalence relation.

Proof. **Reflexivity**: Let $N = (S, T, F, M_{\text{in}}, \ell)$ be a net. The relation

$$\mathscr{R}_{\text{id}} := \{([M,\sigma],[M,\sigma],R^M_{\text{id}}) | M \in \mathbb{N}^S, \sigma \in T^*\}$$

where R^M_{id} is the identity relation on all enabled transitions in M, that is

$$R^M_{\text{id}} := \{(t,t) \mid t \in \text{en}(M)\}.$$

$\mathscr{R}$ is an EPST bisimulation between N and itself, and therefore $N \underline{\leftrightarrow}_{EPST} N$.

Symmetry: For a given EPST-bisimulation $\mathscr{R}$, the relation

$$\mathscr{R}^{-1} := \{([M_1,\sigma_1],[M_0,\sigma_0]R^{-1}) | ([M_0,\sigma_0],[M_1,\sigma_1],R) \in \mathscr{R}\}$$

is also an EPST-bisimulation, where $R^{-1} := \{(v,t) | (t,v) \in R\}$. Therefore $N_0 \underline{\leftrightarrow}_{EPST} N_1 \iff N_1 \underline{\leftrightarrow}_{EPST} N_0$.

Transitivity: For two EPST-bisimulations $\mathscr{R}_0, \mathscr{R}_1$ for the relations $N_0 \underline{\leftrightarrow}_{EPST} N_1$ and $N_1 \underline{\leftrightarrow}_{EPST} N_2$ respectively, we can construct a concatenation

$$\begin{aligned}\mathscr{R}_0;\mathscr{R}_1 := \{([M_0,\sigma_0],[M_2,\sigma_2],R_0;R_1) \mid \\ ([M_0,\sigma_0],[M_1,\sigma_1],R_0) \in \mathscr{R}_0 \\ \wedge([M_1,\sigma_1],[M_2,\sigma_2],R_1) \in \mathscr{R}_1\}\end{aligned}$$

where $R_0;R_1$ is the result of composing the transition relations R_0 and R_1, so $t_0R_0;R_1t_2$ if and only if $\exists t_1.\, t_0R_0t_1 \wedge t_1R_1t_2$.

$\mathscr{R}_0;\mathscr{R}_1$ is an EPST-bisimulation between N_0 and N_2, and therefore

$$N_0 \underline{\leftrightarrow}_{EPST} N_1 \wedge N_1 \underline{\leftrightarrow}_{EPST} N_2 \implies N_0 \underline{\leftrightarrow}_{EPST} N_2.$$

Therefore $\underline{\leftrightarrow}_{EPST}$ is an equivalence relation. □

We can now move onto the proofs that allow us to place EPST-bisimulation in the correct place in the *semantic equivalence lattice*, chiefly that it is finer than both ST- and Ep-bisimulation.

Theorem 4.2. EPST-bisimilarity implies ST-bisimilarity.

Proof. Given an EPST-bisimulation $\mathscr{R}_{\text{EPST}}$ between two nets N_0 and N_1, one can construct an ST-bisimulation by removing the final R relation. Define $\mathscr{R}_{\text{ST}}$ as follows:

$$\mathscr{R}_{\text{ST}} := \{([M_0,\sigma_0],[M_1,\sigma_1]) | ([M_0,\sigma_0],[M_1,\sigma_1],R) \in \mathscr{R}_{\text{EPST}}\}.$$

We now need to verify that $\mathscr{R}_{\text{ST}}$ forms a valid ST-bisimulation. We take each clause of the ST-bisimulation definition in turn and show that it holds:

1. Clause 1 requires that the initial markings are related, and follows directly from EPST-Clause 1.
2. Clause 2 requires that if a transition is enabled in N_0, then N_1 can match it and that the resulting states are also related. This follows from EPST-Clauses 2 and 3, the former guaranteeing that there is a matching transition in N_1, and the latter guaranteeing that the resulting states are related.
3. Clause 3 is the symmetric version of Clause 2, and follows from the symmetry of the EPST-bisimulation definition.
4. Clause 4 requires that if a transition finishes in N_0, then the corresponding transition finishes in N_1 and the resulting states are related. This follows from EPST-Clause 5, which guarantees that the resulting states are related.
5. Clause 5 is the symmetric version of Clause 4, and follows from the symmetry of the EPST-bisimulation definition.
6. Clause 6 requires that if a state is terminal in N_0, then it is also terminal in N_1, and vice versa. This follows from EPST-Clause 6, which guarantees that the two states agree on termination.

Thus, as $\mathscr{R}_{\text{ST}}$ satisfies all the requirements of ST-bisimulation, we have that $N_0 \underline{\leftrightarrow}_{EPST} N_1 \implies N_0 \underline{\leftrightarrow}_{ST} N_1$. $\square$

Theorem 4.3. EPST-bisimulation implies Ep-bisimilarity.

Proof. Given two processes N_0 and N_1 represented by nets such that $N_0 \underline{\leftrightarrow}_{EPST} N_1$ for some EPST-bisimulation $\mathscr{R}_{\text{EPST}}$, we need to show that we can construct an Ep-bisimulation between N_0 and N_1. We construct our candidate relation $\mathscr{R}_{\text{Ep}}$ as follows:

$$\mathscr{R}_{\text{Ep}} := \{(M_0, M_1, R) \mid ([M_0, \epsilon], [M_1, \epsilon], R) \in \mathscr{R}_{\text{EPST}}\}.$$

It just remains to show that $\mathscr{R}_{\text{Ep}}$ satisfies the requirements of Ep-bisimulation.

1. Clause 1 requires that the relation R provides a total matching on all enabled transitions and that it preserves the concurrency structure. This follows from EPST-Clause 2.
2. Clause 2 requires that the initial markings of the nets are related, and follows directly from EPST-Clause 1.
3. Clause 3 requires that if a transition is enabled in N_0, there exists a matching enabled transition in N_1 related by R such that the labels match, that the resulting states are related, and that concurrency is preserved. This follows from EPST-Clause 3, which includes the matching of enabled transitions, the label matching, the resulting states being related, and the preservation of concurrency.
4. Clause 4 follows from the symmetry of the EPST-bisimulation definition.

Therefore, as $\mathscr{R}_{\text{Ep}}$ satisfies all the requirements of Ep-bisimulation, we have that $N_0 \underline{\leftrightarrow}_{EPST} N_1 \implies N_0 \underline{\leftrightarrow}_{Ep} N_1$. $\square$

Now we have established that EPST-bisimulation is an equivalence relation finer than both ST- and Ep-bisimulation, we need to show that it is not overly restrictive or fine. This involves showing that it is the coarsest such bisimulation finer than both bisimulations, meaning that any equivalence relation that is finer than both ST- and Ep-bisimulation must also be finer than or equal to EPST-bisimulation. This is a crucial property for ensuring that EPST-bisimulation is a useful and meaningful equivalence relation, and that it occupies the correct position in the lattice of semantic equivalences.

In order to consider Ep-bisimulations on nets with ST-states, we need to somehow augment Ep-bisimulations defined on nets with regular states to include relations for the markings associated with intermediate ST-states. We do this by defining a new endomorphic function $\mathfrak{I}$ on $\mathbb{N}^{S_0} \times \mathbb{N}^{S_1} \times \mathscr{P}(T_0 \times T_1)$ that takes an Ep-bisimulation $\mathscr{R}$ and returns a new relation to account for these intermediate states.

Definition 4.2 (Intermediate State Operator)**.** Given an Ep-bisimulation

$$\mathscr{R} \subseteq \mathbb{N}^{S_0} \times \mathbb{N}^{S_1} \times \mathscr{P}(T_0 \times T_1),$$

we define the intermediate state operator $\mathfrak{I}$ as follows:

$$\begin{aligned}\mathfrak{I}(\mathscr{R}) := \mathscr{R} &\cup \{(M_0 - {}^*t_0, M_1 - {}^*t_1, R) \mid (M_0, M_1, R) \in \mathscr{R}, t_0 \in \mathrm{en}(M_0) \wedge t_0 R t_1\} \\ &\cup \{(M_0 - {}^*t_0, M_1 - {}^*t_1, R) \mid (M_0, M_1, R) \in \mathscr{R}, t_1 \in \mathrm{en}(M_1) \wedge t_0 R t_1\}.\end{aligned}$$

Or in other words, $\mathfrak{I}(\mathscr{R})$ is $\mathscr{R}$ augmented with additional relations that account for transitions to have begun firing but not yet finished, reusing the same relation R as before they began firing, which although no longer the minimal relation between the two sets of enabled transitions, still forms a valid relation from the perspective of the Ep-bisimulation definition. Observe that $\mathfrak{I}(\mathscr{R})$ still forms a valid Ep-bisimulation, briefly because each relation R is still complete between all enabled transitions, the initial markings are still related via the original relation, and the concurrency preservation clause is still valid as the removed transitions are no longer enabled transitions.

Theorem 4.4. EPST-bisimulation is the coarsest bisimulation finer than both ST- and Ep-bisimulation.

Proof. As we know from Theorem 4.2 and Theorem 4.3, EPST-bisimulation is equivalent to or finer than both ST- and Ep-bisimulation as we can construct valid versions of both from a valid EPST-bisimulation. Therefore, all that remains to be shown is that any equivalence relation finer than both ST- and Ep-bisimulation is necessarily finer than or equal to EPST-bisimulation.

Consider an arbitrary equivalence relation $\sim$ finer than both ST- and Ep-bisimulation, we need to show that $\sim$ is finer or equivalent to $\underline{\leftrightarrow}_{EPST}$. Let N_0 and N_1 be arbitrary nets such that $N_0 \sim N_1$, and let $\mathscr{R}_{\mathrm{Ep}}$ and $\mathscr{R}_{\mathrm{ST}}$ be Ep- and ST-bisimulations between N_0 and N_1 respectively, which are known to exist by the assumption that $\sim$ is finer than both. We can then construct a candidate EPST-bisimulation $\mathscr{R}_{\mathrm{EPST}}$ as follows, a triple $([M_0, \sigma_0], [M_1, \sigma_1], R)$ is in $\mathscr{R}_{\mathrm{EPST}}$ if and only if:

$$([M_0, \sigma_0], [M_1, \sigma_1]) \in \mathscr{R}_{\mathrm{ST}} \wedge (M_0, M_1, R) \in \mathfrak{I}(\mathscr{R}_{\mathrm{Ep}}).$$

We now need to show that $\mathscr{R}_{\text{EPST}}$ forms a valid EPST-bisimulation. We take each clause of the EPST-bisimulation definition in turn and show that it holds:

1. Clause 1 requires that the initial markings are related, and follows directly from the requirements that both Ep- and ST-bisimulations must relate the initial marking.

2. Clause 2 requires that the relation R in a triple $([M_0,\sigma_0],[M_1,\sigma_1],R)$ provides a total matching on all enabled transitions and preserves their concurrency structure. This follows from Clause 1 of the Ep-bisimulation definition, which guarantees the same properties.

3. Clause 3 requires that for a given triple $([M_0,\sigma_0],[M_1,\sigma_1],R) \in \mathscr{R}_{\text{EPST}}$, for every $t_0 \in \text{en}(M_0)$ and $t_1 \in \text{en}(M_1)$ such that $t_0 R t_1$ and the following hold:

   (a) The labels match, this follows from Clause 3 of the Ep-bisimulation definition, which guarantees that the labels of matching transitions are equal.

   (b) There exists a new relation R' such that the resulting ST-state is in the bisimulation, from the definition of $\mathfrak{I}$ as t_0 is enabled and $t_0 R t_1$, we have that $(M_0 - {}^*t_0, M_1 - {}^*t_1, R') \in \mathfrak{I}(\mathscr{R}_{\text{Ep}})$ for some relation R'. By Clause 2 of the ST-bisimulation definition, we also know that $([M_0 - {}^*t_0, \sigma_0 * t_0],[M_1 - {}^*t_1, \sigma_1 * t_1]) \in \mathscr{R}_{\text{ST}}$. Thus by construction of $\mathscr{R}_{\text{EPST}}$, we have that $([M_0 - {}^*t_0, \sigma_0 * t_0],[M_1 - {}^*t_1, \sigma_1 * t_1],R') \in \mathscr{R}_{\text{EPST}}$.

   (c) The new relation R' preserves the old relation for concurrent transitions. This follows directly from Clause 3 of the Ep-bisimulation definition, which guarantees that if tRu and $t \smile\!\bullet\, t_0$, then $tR'u$, and similarly for the other direction.

4. Clause 4 requires that the symmetric version of Clause 2 and Clause 3 hold, which follows from the symmetry of both the Ep- and ST-bisimulation definitions.

5. Clause 5 requires that given a triple $([M_0,\sigma_0 * t_0 * \rho_0],[M_1,\sigma_1 * t_1\rho_1],R) \in \mathscr{R}_{\text{EPST}}$ with $|\sigma_0| = |\sigma_1|$, there must exist some new transition relation R' such that $([M_0 + t_0^*, \sigma_0 * \rho_0],[M_1 + t_1^*, \sigma_1 * \rho_1],R') \in \mathscr{R}_{\text{EPST}}$. By construction we know that

$$(M_0, M_1, R) \in \mathfrak{I}(\mathscr{R}_{\text{Ep}}),$$

   and

$$([M_0,\sigma_0 * t_0 * \rho_0],[M_1,\sigma_1 * t_1 * \rho_1],R) \in \mathscr{R}_{\text{ST}}.$$

   By Clause 4 of the ST-bisimulation definition, we know that

$$([M_0 + t_0^*, \sigma_0 * \rho_0],[M_1 + t_1^*, \sigma_1 * \rho_1]) \in \mathscr{R}_{\text{ST}}.$$

   The state $[M_0,\sigma_0 * t_0 * \rho_0]$ is an intermediate state reached after t_0 began firing. This implies there was a preceding ST-state $[M_0^{\text{pre}},\sigma_0 * \rho_0]$ in which t_0 was enabled and $M_0 = M_0^{\text{pre}} - {}^*t_0$. Similarly, we have a preceding ST-state $[M_1^{\text{pre}},\sigma_1 * \rho_1]$ in which t_1 was enabled and $M_1 = M_1^{\text{pre}} - {}^*t_1$. For the intermediate state to be reachable in the constructed bisimulation, the preceding state must also have been in $\mathscr{R}_{\text{EPST}}$. Thus, for some relation R_{pre}, we must have also had

$$([M_0^{\text{pre}},\sigma_0 * \rho_0],[M_1^{\text{pre}},\sigma_1 * \rho_1],R_{\text{pre}}) \in \mathscr{R}_{\text{EPST}}.$$

This means that $(M_0^{\text{pre}}, M_1^{\text{pre}}, R_{\text{pre}}) \in \mathfrak{I}(\mathscr{R}_{\text{Ep}})$. Now applying Clause 3 of the Ep-bisimulation definition to the state $(M_0^{\text{pre}}, M_1^{\text{pre}}, R_{\text{pre}})$, we know there exists a relation R' such that the resulting markings after the complete firing of t_0 and t_1 are related in $\mathscr{R}_{\text{Ep}}$,

$$(M_0^{\text{pre}} - {}^*t_0 + t_0^*, M_1^{\text{pre}} - {}^*t_1 + t_1^*, R') \in \mathfrak{I}(\mathscr{R}_{\text{Ep}}).$$

By substituting the definition of M_0^{pre} and M_1^{pre}, we then have that

$$(M_0 + t_0^*, M_1 + t_1^*, R') \in \mathscr{R}_{\text{Ep}}.$$

From Clause 4 of the ST-bisimulation definition, we know that

$$([M_0 + t_0^*, \sigma_0 * \rho_0], [M_1 + t_1^*, \sigma_1 * \rho_1]) \in \mathscr{R}_{\text{ST}}.$$

Putting these two statements together, we finally have that

$$([M_0 + t_0^*, \sigma_0 * \rho_0], [M_1 + t_1^*, \sigma_1 * \rho_1], R') \in \mathscr{R}_{\text{EPST}},$$

as required.

6. Clause 6 requires that the two nets agree on termination, that is if $([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R}_{\text{EPST}}$, then $(M_0, \sigma_0) = (\varnothing, \epsilon) \iff (M_1, \sigma_1) = (\varnothing, \epsilon)$. This follows from Clause 6 of the ST-bisimulation definition, and the implicit complete matching of the Ep-bisimulation definition, which guarantees that the two processes agree on termination.

Thus, $\mathscr{R}_{\text{EPST}}$ forms a valid EPST-bisimulation, and therefore $N_0 \leftrightarrow_{EPST} N_1$, which shows that $\sim$ is finer than or equal to EPST-bisimulation. Therefore, EPST-bisimulation is the coarsest bisimulation finer than both ST- and Ep-bisimulation. □

Now we have established EPST-bisimulation's position in the lattice of semantic equivalences relative to ST- and Ep-bisimulation, we can proceed to show that it exhibits the good properties of real-time consistency and justness-preservation. The first of these properties we tackle is justness-preservation.

We introduce the notion of an EPST-bisimulation for specific ST-paths, which is a relation that relates each step of a path in a net to the corresponding step in the other net, whilst also preserving the concurrency structure of the transitions not taken.

Definition 4.3 (EPST bisimulation for paths)**.** Given an EPST-bisimulation $\mathscr{R}$ and two paths $\pi = [M_0, \sigma_0]u_0[M_1, \sigma_1]\ldots$ and $\hat{\pi} = [M_0', \sigma_0']u_0'[M_1', \sigma_1']\ldots$, we write $\pi \mathscr{R} \pi'$ if and only if $l(\pi) = l(\pi')$, and there exist $R_i \subseteq T \times T$ for all $i \in \mathbb{N}$ with $i \leq l(\pi)$ such that

1. $([M_i, \sigma_i], [M_i', \sigma_i'], R_i) \in \mathscr{R}$ for each $i < l(\pi)$,
2. $u_i R_i u_i'$ for each $i < l(\pi)$,
3. if tR_it' and $t \smile\!\!\bullet\, u_i$ with $i < l(\pi)$, then $t' \smile\!\!\bullet\, u_i'$ and $tR_{i+1}t'$, and
4. if tR_it' and $t' \smile\!\!\bullet\, u_i'$ with $i < l(\pi)$, then $t \smile\!\!\bullet\, u_i$ and $tR_{i+1}t'$.

Paths π and π' are EPST-bisimilar, denoted $\pi \underline{\leftrightarrow}_{EPST} \pi'$, if there exists an EPST-bisimulation $\mathscr{R}$ with $\pi\mathscr{R}\pi'$. If $\pi \underline{\leftrightarrow}_{EPST} \pi'$, we also write $\pi\vec{R}\pi'$ if $\vec{R} := (R_0, R_1, \dots)$ are relations that satisfy the above requirements.

With this structure in mind, we can now prove the lemma that related paths are concurrent to a given transition if and only if the corresponding path in the other net is also concurrent to the same transition.

Lemma 4.1. If $\pi\vec{R}\rho$ with π finite and tR_0t', then $t \smile\!\!\bullet\, \hat{\pi} \iff t' \smile\!\!\bullet\, \hat{\rho}$.

Proof. By using the symmetry of the statement, we need only prove one direction, we approach the "only-if" direction via induction on the length of the finite path π. The base case when $l(\pi) = 1$, or $\hat{\pi} = (u_0)$, holds by direct application of Clause 4 of the definition of EPST-bisimulation for paths. Then suppose $\pi u[M,\sigma]\vec{R}\rho u'[M',\sigma']$ where $l(\pi) = n$, we know that $t' \smile\!\!\bullet\, \hat{\rho} \implies t \smile\!\!\bullet\, \hat{\pi}$ by induction and that $t' \smile\!\!\bullet\, u' \implies t \smile\!\!\bullet\, u$ by the base case, putting these together $t' \smile\!\!\bullet\, \hat{\rho}u' \implies t \smile\!\!\bullet\, \hat{\pi}u$, as required. □

Finally, we can use the previous definition and lemma to show that EPST-bisimulation preserves justness. This means that if two paths are EPST-bisimilar, then they are both just or unjust with respect to the same set of blocking actions.

Theorem 4.5. EPST-bisimulation preserves justness. That is, given two paths π and π' such that $\pi \underline{\leftrightarrow}_{EPST} \pi'$, and a set of blocking actions B, then π if B-just if and only if π' is B-just.

Proof. Let $\pi = [M_0,\sigma_0]u_0[M_1,\sigma_1]u_1\dots$ and $\pi' = [M'_0,\sigma'_0]u'_0[M'_1,\sigma'_1]u'_1\dots$ such that $\pi \underline{\leftrightarrow}_{EPST} \pi'$ for some EPST-bisimulation $\mathscr{R}$ and relations $\vec{R}$ according to Definition 4.3. Suppose π is B-unjust for some set of blocking visible actions B, and we have some visible $t \in T$ such that $\ell(t) \notin B$ and ${}^*t \subseteq M_i$ such that $t \smile\!\!\bullet\, \hat{\rho}$ for each finite prefix ρ of the suffix $[M_i,\sigma_i]u_i[M_{i+1},\sigma_{i+1}]u_{i+1}\dots$ of π. By the symmetry of the statement, it suffices to prove the contrapositive of one direction, in this case that π' is also B-unjust. Pick a visible $t' \in T$ such that tR_it', which is guaranteed to exist by Clause 2 of the EPST-bisimulation definition. Now ${}^*t' \subseteq M'_i$ as it must also be enabled, and by the same clause of the EPST-bisimulation definition $t' \in T$ and $\ell(t') = \ell(t) \notin B$. We now only need that $t' \smile\!\!\bullet\, \hat{\rho}'$ for each finite prefix ρ' of π' formed in the same way as ρ above, this follows from Lemma 4.1. □

Moving onto real-time consistency, this is a comparatively simpler proof, and follows directly from the fact that EPST-bisimulation is finer than ST-bisimulation.

Theorem 4.6. EPST-bisimulation is real-time consistent.

Proof. EPST-bisimilarity implies ST-bisimilarity by application of Theorem 4.2, and ST-bisimulation is real-time consistent [11, Theorem 7.2, p. 17], therefore $\underline{\leftrightarrow}_{EPST}$ is real-time consistent. □

Chapter 5

Congruence Properties of EPST-Bisimulation

5.1 The Concept of Congruence

In the context of equivalence relations, an equivalence relation $\sim$ is said to be a *congruence* for an operator, or that the operator is *compositional* for the equivalence, if the equivalence is preserved under the application of the operator. This means that if two processes are equivalent under the relation, then applying a compositional operator to both processes will yield two new processes that are also equivalent under the relation.

Formally, for a language L, we define this in terms of *contexts*, where a context is an L-expression with a hole, denoted $C[\ ]$. For example $C[\ ] = A + [\ ]$, and $C[P]$ is the result of substituting P into the hole of $C[\ ]$.

Definition 5.1 (Congruence)**.** An equivalence relation $\sim$ is a congruence for a language L if $P \sim Q \implies C[P] \sim C[Q]$ for every context $C[\ ]$.

We say that an equivalence relation is a congruence for an entire language L if it is a congruence for every operator in L. In this chapter, we will show that EPST-bisimulation is a congruence for the ACP (see Section 2.2). This is an important property, as it allows us to iteratively build larger processes from smaller ones whilst preserving their equivalence under EPST-bisimulation, particularly useful in the context of process algebra.

5.2 Congruence Proofs for ACP Operators

5.2.1 Root-unwinding

In order to properly apply the semantics of ACP operators on nets, we need to first define the root-unwinding operator. This is due to the fact that the semantics of many ACP operators, such as the choice operator $+$, rely on the creation of new initial markings for the processes involved that flow in one direction, and therefore require no incoming edges to the initial places.

As we are utilising the definition of the semantics of the ACP operators on nets from Van Glabbeek and Vaandrager [11], we need to introduce the *root-unwinding operator*. This is because many of the definitions of operators, such as the choice operator $+$, require that we are working on nets with *acyclic roots*. We say that a net has acylic roots if all places in its initial marking have no incoming edges.

Definition 5.2 (Root unwinding)**.** The root-unwinding map $\mathfrak{R}$ is defined as follows [11, p. 2.10.1]. Let $N_0 \in \mathbb{N}_\lambda(A)$ be a net over some action set A, and let $M_{\text{cyc}} = \{s \in M_0^{\text{in}} \mid {}^*s \neq \varnothing\}$ be the set of cyclic root elements, and let $M_{\text{cyc}}^c = \{s^e \mid s \in M_{\text{cyc}}\}$ be a copy of this set. Then $\mathfrak{R}(N_0)$ is the set N_1 obtained by adding the places in M_{cyc}^c and put them in the initial marking instead of the elements of M_{cyc}:

$$S_1 = S_0 \cup M_{\text{cyc}}^c \quad \text{and} \quad M_1^{\text{in}} = (M_0^{\text{in}} - M_{\text{cyc}}) \cup M_{\text{cyc}}^c.$$

We define a set $\mathbb{U}$ by

$$\mathbb{U} = \{M_{\text{cyc}} - (\bigcup_{t \in X} {}^*t) \mid X \text{ is a finite subset of } T_0\}$$

n.b. the cardinality of $\mathbb{U}$ is less than λ, for each $U \in \mathbb{U}$ and $t \in T_0$ such that ${}^*t \cap U$ is non-empty, introduce a new transition $\langle U, t\rangle$ such that

$${}^*\langle U, t\rangle = ({}^*t - U) \cup \{s^c \mid s \in {}^*t \cap U\} \quad \text{and} \quad \langle U, t\rangle^* = t^*.$$

The label of $\langle U, t\rangle$ is also the label of t. Because the cardinality of $\mathbb{U}$ is less than λ, the number of new transitions in the root-unwinding N_1 is also less than λ. Thus we avoid the cardinality problem that arises if we introduce a new transition $\langle U, t\rangle$ for every subset U of M_{cyc}.

As the source paper is concerned with the implications for potentially infinite nets and the associated cardinality problems, this definition is formal and complex. The intuitive idea adequate for understanding of this paper is that the root-unwinding operator creates an additional place for every place in the initial marking with cyclic roots, modifies the initial marking so that this new place takes the place of the old cyclic place, and adds new edges to the flow relation to connect each new acylic initial place to its old cyclic place.

5.2.2 Proofs of Congruence of EPST-Bisimulation on ACP

We are now in a position to prove congruence properties of the EPST-bisimulation with respect to the ACP operators, starting with the choice operator $+$.

Lemma 5.1. EPST-bisimilarity is a congruence for the choice operator $+$.

Proof. Let N_0, N_1, and P be arbitrary processes represented by Petri nets. We need to show that $N_0 \leftrightarroweq_{EPST} N_1 \implies N_0 + P \leftrightarroweq_{EPST} N_1 + P$. Let $\tilde{N}_i = \mathfrak{R}N_i$, $\tilde{P} = \mathfrak{R}P$, and $N_i' := \tilde{N}_i + \tilde{P}$ for $i \in \{0, 1\}$ where $\mathfrak{R}$ is the root-unwinding operator, N_i' is then defined as follows [8]:

- $S_i' := (\tilde{S}_i - \tilde{M}_i^{\text{in}}) \cup (\tilde{S}_P - \tilde{M}_P^{\text{in}}) \cup \tilde{M}_i^{\text{in}} \times \tilde{M}_P^{\text{in}}$

- $T'_i := \tilde{T}_i \cup \tilde{T}_P$
- $F'_i := ((\tilde{F}_i \cup \tilde{F}_P) \cap (S'_i \times T'_i \cup T'_i \times S'_i)) \cup \{((s_0, s_1), t) \mid (s_0, t) \in \tilde{F}_i \vee (s_1, t) \in \tilde{F}_P\}$
- $M_i^{\text{in}\prime} = \tilde{M}_i^{\text{in}} \times \tilde{M}_P^{\text{in}}$
- $\ell'_i = \tilde{\ell}_i \cup \tilde{\ell}_P$

Intuitively, this is the result of combining the two nets, and replacing the initial places of both with a cartesian product of the initial places. The result of which is that if at least one token is consumed by either process, then only that process will have the presets of its initial transitions satisfied and can continue, with the other one being starved of tokens, resulting in a logical choice between the two.

Let $\mathscr{R}$ be an EPST-bisimulation between N_0 and N_1, and $\mathscr{R}_{\text{id}}$ be the identity EPST-bisimulation between P and itself. We are now in a position to define our candidate EPST-bisimulation $\mathscr{R}'$ as follows, a triple $([M'_0, \sigma'_0], [M'_1, \sigma'_1], R')$ is in $\mathscr{R}'$ if and only if one of the following conditions holds:

(a) The system is in its initial state, that is $M'_0 = M_0^{\text{in}\prime}$ and $M'_1 = M_1^{\text{in}\prime}$ and $\sigma'_0 = \sigma'_1 = \epsilon$, and the relation R' is the union of the initial relations from the two source bisimulations. By definition, there exists some relation R such that

$$([M_0^{\text{in}}, \epsilon], [M_1^{\text{in}}, \epsilon], R) \in \mathscr{R},$$

and a relation R_{id} such that

$$([M_P^{\text{in}}, \epsilon], [M_P^{\text{in}}, \epsilon], R_{\text{id}}) \in \mathscr{R}_P,$$

and therefore $R' = R \cup R_{\text{id}}$.

(b) A choice has been made, and the new processes are either both evolving as N_0 and N_1 or as P. We can therefore split based on this choice:

- N-phase, the marking M'_0 can be expressed as some marking of N_0, say M_0, and some partial marking of the initial states M_0^{partial}, such that $M'_0 = M_0 + M^{\text{partial}}$, and similarly for M'_1 for the same M^{partial} such that $M'_1 = M_1 + M^{\text{partial}}$. In this case, the relation R' must be contained within the original relation $\mathscr{R}$:

$$([M_0, \sigma_0], [M_1, \sigma_1], R') \in \mathscr{R}.$$

- P-phase, the marking M'_0 can be expressed as some marking of P, say M_P, and some partial marking of the initial states M_P^{partial}, such that $M'_0 = M_P + M^{\text{partial}}$, and similarly for M'_1 for the same M^{partial} such that $M'_1 = M_P + M^{\text{partial}}$. In this case, the relation R' must be the identity relation on the enabled transitions of M_P:

$$([M_P, \sigma_0], [M_P, \sigma_1], R') \in \mathscr{R}_P.$$

We can now verify that $\mathscr{R}'$ forms a valid EPST-bisimulation.

1. Clause 1 requires that the initial markings of the two nets are related, which follows directly from the construction of $\mathscr{R}'$.

2. Clause 2 requires that the relation R' provides a total matching on all enabled transitions and preserves their concurrency structure. This follows from the fact that the relation R is a total matching on all enabled transitions of N_0 and N_1, and the relation R_{id} is a total matching on all enabled transitions of P, and therefore their union is also a total matching. The concurrency preservation also holds: any transitions from the disjoint components are concurrent by construction, so the equivalence holds trivially. For any two transitions within the same component, the property is inherited directly from the appropriate original relation.

3. Clause 3 requires that for a given triple $([M_0', \sigma_0'], [M_1', \sigma_1'], R') \in \mathscr{R}'$, for every $t_0' \in \text{en}(M_0')$ there exists a matching $t_1' \in \text{en}(M_1')$ such that $t_0' R' t_1'$ such that a number of key properties hold. By construction $R' = R \cup R_{\text{id}}$ where R is the transition relation from $\mathscr{R}$ and R_{id} is the transition relation from $\mathscr{R}_{\text{P}}$. There are two possible sources for the transition t_0'

   - If $t_0' \in T_0$ then we are in the N-phase, and therefore there must exist a $t_1' \in T_1$ such that $t_0' R t_1'$, and the key properties hold by assumption that $\mathscr{R}$ is an EPST-bisimulation.

   - If $t_0' \in T_P$ then we are in the P-phase, and therefore $t_0' = t_1'$ and $t_0' R_{\text{id}} t_1'$, and the key properties hold by construction of $\mathscr{R}_P$.

   Therefore, in either case, the key properties hold, and thus the entire clause holds.

4. Clause 4 holds by the symmetry of the definition of $\mathscr{R}'$ and the underlying symmetry of the two bisimulations $\mathscr{R}$ and $\mathscr{R}_P$.

5. Clause 5 requires that given a triple $([M_0', \sigma_0' * t_0' \rho_0'], [M_1', \sigma_1' * t_1' \rho_1'], R') \in \mathscr{R}'$, there exists some relation R'' such that $([M_0' + t_0'^*, \sigma_0' * \rho_0'], [M_1' + t_1'^*, \sigma_1' * \rho_1'], R'') \in \mathscr{R}'$. This follows from the fact that if t_0' is in the N-phase, then there exists a matching t_1' in the N-phase such that $([M_0', \sigma_0'] + t_0'^*, [M_1', \sigma_1'] + t_1'^*) \in \mathscr{R}$, and therefore the clause holds. If t_0' is in the P-phase, then we know that $([M_P, \sigma_P] + t_P^*, [M_P, \sigma_P] + t_P^*) \in \mathscr{R}_P$, and therefore the clause holds.

6. Clause 6 requires that the two nets agree on termination, that is if

   $$([M_0', \sigma_0'], [M_1', \sigma_1'], R') \in \mathscr{R}',$$

   then

   $$(M_0', \sigma_0') = (\varnothing, \epsilon) \iff (M_1', \sigma_1') = (\varnothing, \epsilon).$$

   By construction, if one net is in a N-phase, then the other net is also in an N-phase, and the clause holds by assumption that $\mathscr{R}$ is an EPST-bisimulation. If both nets are in a P-phase, then they are both in the same state, and therefore the clause holds by construction of $\mathscr{R}_P$.

Thus, as $\mathscr{R}'$ is a valid EPST-bisimulation between N_0' and N_1', $N_0 + P \mathrel{\underline{\leftrightarrow}}_{EPST} N_1 + P$ as required. □

Lemma 5.2. EPST-bisimilarity is a congruence for the sequential composition operator .

Proof. Let $N_0 = (S_0, T_0, F_0, M_0^{\text{in}}, \ell_0)$, $N_1 = (S_1, T_1, F_1, M_1^{\text{in}}, \ell_1)$, and $P = (S_P, T_P, F_P, M_P^{\text{in}}, \ell_P)$ be arbitrary processes represented by Petri nets. We need to show that $N_0 \leftrightarrow_{EPST} N_1 \implies N_0.P \leftrightarrow_{EPST} N_1.P$. Let $\tilde{N}_0 = \mathfrak{R}N_0$, $\tilde{N}_1 = \mathfrak{R}N_1$, and $\tilde{P} = \mathfrak{R}P$.

The sequential composition $N_i' := \tilde{N}_i.\tilde{P}$ is defined in terms of complementary places as follows [11, p. 3.4.1]:

$$\begin{aligned} S_i' &:= S_i \cup (\tilde{S}_P - \tilde{M}_P^{\text{in}}) \cup (S_i \times \tilde{M}_P^{\text{in}}) \\ T_i' &:= T_i \cup \tilde{T}_P \\ (M_i^{\text{in}})' &:= M_i^{\text{in}} \cup (S_i - M_i^{\text{in}}) \times \tilde{M}_P^{\text{in}} \end{aligned}$$

The flow relation F_i' is defined as the union of the original flow relation F_i and $\tilde{F}_P \cap (S_i' \times T_i' \cup T_i' \times S_i')$ with the following additional transitions:

$$\begin{aligned} &[(s,t) \in F_i \wedge (t,s) \notin F_i \wedge s' \in \tilde{M}_P^{\text{in}}] \to (t,(s,s')) \in F_i' \\ &[(t,s) \in F_i \wedge (s,t) \notin F_i \wedge s' \in \tilde{M}_P^{\text{in}}] \to ((s,s'),t) \in F_i' \\ &[(s_0,s_1) \in S_i \times \tilde{M}_P^{\text{in}} \wedge (s_1,t) \in \tilde{T}_P] \to ((s_0,s_1),t) \in F_i' \end{aligned}$$

Or the original arrows with connecting additional arrows from the terminating places of N_i to the complementary places, and from the complementary places to the initial places of P, unwound where appropriate. Finally, the labelling relation is defined as:

$$\ell_i' := \ell_i \cup \ell_P$$

We are now in a position to construct a candidate EPST-bisimulation $\mathscr{R}'$ between N_0' and N_1'. A state in the composed nets can be in one of two phases: either the first net N_i is running, or it has terminated and the second net R is running. Let $([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R}'$ if and only if one of the following conditions holds:

1. The entire triple is contained within the original relation $\mathscr{R}$.
2. All of the following hold, $M_0 = M_1 \subseteq S_P$, $\sigma_0 = \sigma_1 \in T_P^*$, and R is the identity relation on the enabled transitions of M_0 and M_1.
3. M_0 can be decomposed into two disjoint markings $M_0' \in \mathbb{N}^{S_0}$ and $M_{\text{comp}} \in \mathbb{N}^{S_P}$ such that $M_0 = M_0' + M_{\text{comp}}$ and $M_1 = M_1' + M_{\text{comp}}$ where $M_1' \in \mathbb{N}^{S_1}$ and M_{comp} are also disjoint, that $\sigma_0 \in T_0^*$ and $\sigma_1 \in T_1^*$, and $([M_0', \sigma_0], [M_1, \sigma_1'], R) \in \mathscr{R}$.

Or in other words, $\mathscr{R}'$ contains the original relation $\mathscr{R}$ in addition to associating all partially completed states of N_0 and N_1 and R with itself. We are now in a position to verify that $\mathscr{R}'$ forms a valid EPST-bisimulation between N_0' and N_1'. For each triple $([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R}'$, we consider its origin to show it satisfies the clauses of the EPST-bisimulation definition.

1. If the triple is sourced from the original relation $\mathscr{R}$, then it satisfies all the clauses of the EPST-bisimulation definition by assumption.

2. If the triple is sourced from the identity relation, then it trivially satisfies all the clauses of the EPST-bisimulation definition, as it relates every enabled transition to itself, and the initial markings are related by construction, see the proof of Theorem 4.1.

3. If the triple is sourced from the third clause, then we know that M_0 and M_1 can be decomposed into two sets disjoint with a set containing some marking on the complementary termination places of N_0 and N_1 respectively, call these M_0' and M_1'. We know that M_0' and M_1' are related alongside σ_0 and σ_1 in $\mathscr{R}$, and therefore that these aspects satisfy the requirements for the EPST-bisimulation definition. The remaining aspect of the two markings is then related by the extension of the relation R' with the identity relation, and therefore satisfies the requirements of the EPST-bisimulation definition.

Therefore, as $\mathscr{R}'$ satisfies all the clauses of the EPST-bisimulation definition, it is a valid EPST-bisimulation between N_0' and N_1', and thus $N_0' \underline{\leftrightarrow}_{EPST} N_1'$ as required. □

Lemma 5.3. EPST-bisimilarity is a congruence for the parallel composition operator $||$

Proof. Let N_0, N_1, and P be arbitrary Petri nets such that $N_0 \underline{\leftrightarrow}_{EPST} N_1$ for some EPST-bisimulation $\mathscr{R}$. We need to show that

$$N_0||P \underline{\leftrightarrow}_{EPST} N_1||P.$$

For some communication function $\gamma : A \times A \to A$. Taking the semantics of the parallelisation operator for petri nets [11, p. 3.5], define $N_i' := N_i||P, i \in \{0,1\}$ as the disjoint union of the components N_i and P, plus additional transitions for communication.

- $S_i' = S_i \cup S_P$;
- $T_i' = T_i \cup T_P \cup \{(t,u) \mid t \in T_i, u \in T_P, \gamma(\ell_i(t), \ell_P(u)) \neq \delta\}$;
- $M_i^{\text{in}\prime} = M_i^{\text{in}} \cup M_P^{\text{in}}$;
- For a communication transition (t,u), its preset is ${}^*(t,u) = {}^*t \cup {}^*u$ and its postset is $(t,u)^* = t^* \cup u^*$. The label is $\ell_i'((t,u)) = \gamma(\ell_i(t), \ell_P(u))$.

Let $\mathscr{R}_{\text{id}}$ be the identity EPST-bisimulation between P and itself, which is guaranteed to exist as $\underline{\leftrightarrow}_{EPST}$ forms an equivalence relation. We can now define our candidate EPST-bisimulation $\mathscr{R}'$ between N_0' and N_1' as follows, a triple $([M_0', \sigma_0'], [M_1', \sigma_1'], R_{||})$ is in $\mathscr{R}'$ if and only if there exists a relation $R \subseteq T_0 \times T_1$ such that the following conditions hold:

(a) The marking M_0' can be decomposed into disjoint markings $M_0 \in \mathbb{N}^{S_0}$ and $M_P \in \mathbb{N}^{S_P}$ such that $M_0' = M_0 + M_P$, and M_1' can be decomposed into disjoint markings $M_1 \in \mathbb{N}^{S_1}$ and the same M_P such that $M_1' = M_1 + M_P$.

(b) The firing sequences are the same length, and their transitions correspond position-wise as follows, for each position j with the j-th element of σ_0' denoted x_j and the j-th element of σ_1' denoted y_j:

  - If $x_j \in T_0$ then $y_j \in T_1$ and $x_j R y_j$

- If $x_j \in T_P$ then $y_j = x_j$
- If $x_j = (t_0, u)$ and $\gamma(\ell_0(t_0), \ell_P(u)) \neq \delta$, then $y_j = (t_1, u)$ with $t_0 R t_1$

(c) The filtered versions of the firing sequences are contained within the original bisimulations. Let $\pi_i(\sigma)$ for $i \in \{0, 1, P\}$ represent σ filtered to only contain transitions from original process i, then we require that

$$([M_0, \pi_0(\sigma_0')], [M_1, \pi_1(\sigma_1')], R) \in \mathscr{R} \wedge ([M_P, \pi_P(\sigma_0')], [M_P, \pi_P(\sigma_1')], R_{\text{id}}) \in \mathscr{R}_{\text{id}}$$

where R_{id} is the identity relation on all enabled transitions in M_P.

(d) The relation $R_{||} \subseteq T_0' \times T_1'$ is derived from R as follows:

- $t_0 R_{||} t_1 \iff t_0 R t_1$.
- $u \in \text{en}(M_P) \implies u R_{||} u$.
- $(t_0, u) R_{||} (t_1, u) \iff t_0 R t_1$ and $\gamma(\ell_0(t_0), \ell_P(u)) \neq \delta$.

We are now in a position to prove that $\mathscr{R}'$ forms a valid EPST-bisimulation.

1. Clause 1 requires that the initial markings $M_0^{\text{in}\prime}$ and $M_1^{\text{in}\prime}$ with the empty firing sequence ϵ are related by some relation $R_{||}$. We can decompose $M_i^{\text{in}\prime}$ into disjoint multisets M_i^{in} and M_P^{in}, by assumption there exists a relation R such that $([M_0^{\text{in}}, \epsilon], [M_1^{\text{in}}, \epsilon], R) \in \mathscr{R}$ and therefore there exists a $R_{||}$ as specified such that $([M_0^{\text{in}\prime}, \epsilon], [M_1^{\text{in}\prime}, \epsilon], R_{||}) \in \mathscr{R}'$.

2. Clause 2 requires that given $([M_0', \sigma_0'], [M_1', \sigma_1'], R_{||} \in \mathscr{R}'$, the relation $R_{||}$ must form a complete relation on all enabled transitions in M_0' and M_1'. By construction of the relation $R_{||}$, we know that $R_{||}$ is a complete relation on all enabled transitions in M_0 and M_1 as R is assumed to be a complete relation, and that $R_{||}$ is the identity relation on all enabled transitions in M_P. Therefore, Clause 2 holds.

3. Clause 3 requires that given a triple $([M_0', \sigma_0'], [M_1', \sigma_1'], R_{||}) \in \mathscr{R}'$, for each transition t_0' enabled in M_0' there exists a transition t_1' enabled in M_1' such that a number of properties hold. By construction of $R_{||}$ there are three distinct possibilities:

   - If $t_0' \in T_0$ then $t_0' R t_1'$, and thus all the required properties are inherited from the original relation R from $\mathscr{R}$.
   - If $t_0' \in T_P$ then $t_0' = t_1'$, and the properties hold trivially as the transition is the same in both nets.
   - If $t_0' = (t_0, u), t_1' = (t_1, u)$ for some transitions $t_0 \in T_0, t_1 \in T_1$, and $u \in T_P$, then $t_0 R t_1$. By definition the labels must match, and therefore

   $$\ell_0(t_0) = \ell_1(t_1) = \gamma(\ell_0(t_0), \ell_P(u)).$$

   The pre- and post-sets of the communication transition are defined as the union of the pre- and post-sets of the original transitions, and therefore all the regular properties hold.

4. Clause 4 follows by the symmetry of the EPST-bisimulation definition and the symmetric construction of the relation $R_{||}$.

5. Clause 5 requires that given a triple with some in-progress transitions t'_0 and t'_1, $([M'_0, \sigma'_0 * t'_0 * \rho'_0], [M'_1, \sigma'_1 * t'_1 * \rho'_1], R_{||}) \in \mathscr{R}'$, then the ST-states after t'_0 and t'_1 finish must also be related in $\mathscr{R}'$ for some $R'_{||}$. There are three possibilities for the transitions t'_0 and t'_1:

   - If $t'_0 \in T_0$ then $t'_1 \in T_1$, and the property is inherited directly from the original bisimulation $\mathscr{R}$.
   - If $t'_0 \in T_P$, then $t'_0 = t'_1$, and the property holds trivially.
   - If t'_0 is some synchronised communication action $t'_0 = (t_0, u)$ for some $t_0 \in T_0$ and $u \in T_P$, then $t'_1 = (t_1, u)$. By construction of the communication transitions, the pre- and post-sets of the communication transition are the unions of the pre- and post-sets of the original transitions, and therefore the ST-states after t'_0 and t'_1 finish are related in $\mathscr{R}'$.

   Therefore, Clause 5 holds.

6. Clause 6 requires that the two nets agree on termination, that is if

$$([M'_0, \sigma'_0], [M'_1, \sigma'_1], R_{||}) \in \mathscr{R},$$

   then

$$M'_0 = \varnothing \wedge \sigma'_0 = \epsilon \iff M'_1 = \varnothing \wedge \sigma'_1 = \epsilon.$$

   By construction of $\mathscr{R}'$ we can decompose M'_0 and M'_1 into disjoint markings M_0, M_1 and a P-marking M_P. By construction of $\mathscr{R}$ and $\mathscr{R}_P$ we know that

$$M_0 = \varnothing \wedge \sigma'_0 = \epsilon \iff M_1 = \varnothing \wedge \sigma'_1 = \epsilon.$$

   and M_P is the empty set if and only if M_P is the empty set is trivially true. Therefore, the two nets agree on termination, and Clause 6 holds.

Thus $\mathscr{R}'$ forms a valid EPST-bisimulation between N'_0 and N'_1, and therefore $N'_0 \leftrightarroweq_{EPST} N'_1$ as required. □

Lemma 5.4. EPST-bisimilarity is a congruence for the encapsulation operator ∂_H for some set of actions H.

Proof. Let N_0 and N_1 be nets such that $N_0 \leftrightarroweq_{EPST} N_1$ for some EPST-bisimulation $\mathscr{R}$, and let H be a set of visible actions to encapsulate. We need that $N_0 \leftrightarroweq_{EPST} N_1 \implies \partial_H(N_0) \leftrightarroweq_{EPST} \partial_H(N_1)$.

By the definition of the semantics of ACP on Petri nets, $N'_i := \partial_H(N_i), i \in \{0, 1\}$ is defined:

- $S'_i := S_i$
- $T'_i := \{t \in T_i \mid \ell_i(t) \notin H\}$
- F'_i is the relation F_i restricted to the places in S_i and transitions in T'_i

- $M_i^{\text{in}} := M_i^{\text{i}}$
- ℓ_i' is the labelling function ℓ_i restricted to the domain T_i'

We can now construct the candidate EPST-bisimulation $\mathscr{R}'$ for N_0' and N_1' by restricting $\mathscr{R}$:

$$\mathscr{R}' := \{([M_0,\sigma_0],[M_1,\sigma_1],R') \in \mathscr{R} \mid \sigma_0 \in (T_0')^* \wedge \sigma_1 \in (T_1')^* \wedge R' \subseteq T_0' \times T_1'\}$$

Essentially taking $\mathscr{R}$ and discarding any relations that involve a transition removed by ∂_H, either in the firing sequences or in the transition relation. We now need to verify that $\mathscr{R}'$ is still an EPST-bisimulation, we treat the clauses from Definition 4.1 in order:

1. By assumption that $\mathscr{R}$ is a valid EPST-bisimulation, we know that $([M_0^{\text{in}},\epsilon],[M_1^{\text{in}}],R) \in \mathscr{R}$ for some relation R. The initial markings of N_0' and N_1' are unchanged, and the empty firing sequence ϵ is still valid in $(T_0')^*$ and $(T_1')^*$. The initial transition relation is $R' = R \cap (T_0' \times T_1')$. Therefore the triple $([M_0^{\text{in}},\epsilon],[M_1^{\text{in}},\epsilon],R') \in \mathscr{R}'$.
2. Assume that we have $([M_0,\sigma_1],[M_1,\sigma_1],R') \in \mathscr{R}'$ and there is a transition $t_0 \in T_0'$ enabled in the marking M_0. Since $T_0' \subseteq T_0$ and $\mathscr{R}' \subseteq \mathscr{R}$, this configuration corresponds to a valid configuration in the original bisimulation. Because $N_0 \leftrightarrow_{EPST} N_1$ there must be a matching transition $t_1 \in T_1$ such that $\ell_0(t_0) = \ell_1(t_1)$ and a new state $([M_0 - {}^*t_0, \sigma_0 * t_0],[M_1 - {}^*t_1,\sigma_1 * t_1],R'') \in \mathscr{R}$. Since $t_0 \in T_0'$, we know its label $\ell_0(t_0) \notin H$. Because the labels must match, $\ell_1(t_1) \notin H$, which implies that t_1 wasn't removed by the encapsulation operator, and therefore $t_1 \in T_1'$. Thus all the components of the resulting state are restricted to the transitions remaining in N_0' and N_1', and so the resulting state is in $\mathscr{R}'$.
3. Symmetric.
4. The encapsulation operator does not interfere with the preset or postset of any remaining transitions, and therefore this property is inherited directly by $\mathscr{R}'$ from $\mathscr{R}$
5. A triple $([M_0,\sigma_0],[M_1,\sigma_1],R') \in \mathscr{R}'$ was necessarily also in $\mathscr{R}$, and therefore we inherit the property that $(M_0,\sigma_0) = (\varnothing,\epsilon) \iff (M_1,\sigma_1) = (\varnothing,\epsilon)$.

Therefore, as all the conditions are met, $\mathscr{R}'$ is a valid EPST-bisimulation between $\partial_H(N_0)$ and $\partial_H(N_1)$, thus $\leftrightarrow_{EPST}$ is a congruence for the ACP encapsulation operator. □

Lemma 5.5. EPST-bisimilarity is a congruence for the abstraction operator τ_I for a set of visible actions I.

Proof. Given two nets N_0 and N_1 over a set of visible actions A such that $N_0 \leftrightarrow_{EPST} N_1$ and a set of visible actions $I \subseteq A$ to abstract from, we claim that any EPST-bisimulation between N_0 and N_1 is also an EPST-bisimulation between $\tau_I(N_0)$ and $\tau_I(N_1)$.

The abstraction operator renames all the actions in the set I to the internal action τ, in the context of Petri nets this means simply that the original labelling function ℓ is modified to create ℓ' as follows:

$$\ell'(t) = \begin{cases} \tau, & \text{if } t \in I \\ \ell(t), & \text{otherwise.} \end{cases}$$

This therefore means that all aspects of the EPST-bisimulation are untouched except for the label matching sub-clause of Clause 3. In this case, if two transitions are related by a transition relation R in the original bisimulation, their labels must match, meaning that they are both either renamed to τ or are left alone, meaning their labels continue to match. Therefore any EPST-bisimulation between N_0 and N_1 is also a valid EPST-bisimulation between $\tau_I(N_0)$ and $\tau_I(N_1)$ without modification. □

Theorem 5.1. EPST-bisimilarity is a congruence for the language ACP.

Proof. By application of Lemmas 5.1, 5.2, 5.3, 5.4, and 5.5. □

Chapter 6

Action Refinement and EPST-Bisimulation

Having established congruence under the ACP operators, we now extend the argument to *action refinement*. To this end we define the notion of **action refinement** and provide a limited definition of its semantics on Petri nets. We then prove that EPST-bisimulation is preserved under action refinement in a proof of a similar, but more complex, style to the congruence proofs of the previous chapter.

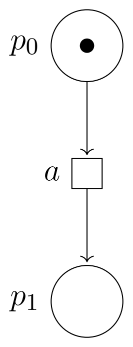
6.1 Action Refinement

Definition 6.1. A **refinement** in the context of Petri nets over a set of visible actions A is a function $r : A \to \mathbb{N}(A)$ that maps visible actions to well-behaved Petri nets; we define a well-behaved Petri net to mean one that:

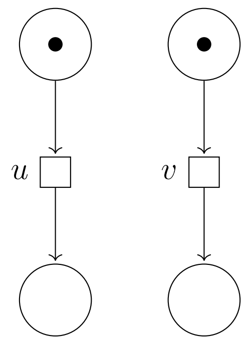
1. Is finite;
2. Contains one input place, that is only one place with no incoming edges, and the initial marking consists of precisely one token in this place;
3. Contains one termination place, that is one place all paths of execution terminate at, that has no outgoing edges;
4. Is deadlock-free, meaning every reachable marking (prior to the termination place) has at least one enabled transition.

This definition of well-behaved essentially restricts the nets one can replace individual transitions with to ones that "act like" single transitions from a macro perspective. To see why this is important, consider the following net that contains a single action a and a refinement net to be mapped to it.

Now in order to replace the single a transition with this new net we face several challenges, we need to somehow get two tokens from a single token, which we could achieve via a τ transition with two outgoing edges. But more than that, we need two tokens that are causally independent, as $u \smile v$, something that simply cannot be created



(a) Trivial net with one a transition



(b) Net defined for $r(a)$

from the single token in p_0 without invalidating the concurrency-preserving aspects we aim to preserve. It is for this reason that we restrict the possible refinement sub-nets to those with a single parallel component.

We also restrict refinements to not include deadlock states, as these change the macro-behaviour of the net. If we have a single transition a, and it begins to fire, then eventually under the minimal assumption of progress it will finish and we will move into its post-set state. If we allow this transition a to be replaced with a refinement that contains a deadlock, and the refined net ends up in this state, then the token that entered the refinement sub-net will never leave, and from a macro-perspective a will never finish firing.

6.2 Refinement on Petri Nets

We are now in a position to utilise these refinements on Petri nets as a whole, we begin by introducing the process we utilise for applying a refinement to a Petri net, then the process for *de-refining* a given state to the equivalent state in the unrefined net, and then briefly argue for the restrictions and additional steps we introduce.

Definition 6.2. Given a Petri net $N = (S, T, F, M^{\text{in}}, \ell)$ over some set of visible actions A and a refinement r defined on all actions in A, we say $r(N)$ to refer to the Petri net with r applied throughout, we construct this net as follows, for every $t \in T$:

1. Remove t from T and all edges in F involving t.
2. Add the places, transitions, and edges of $r(\ell(t))$ save for the initial and termination place (and associated edges) into the net with all transitions t' in the sub-net renamed to (t, t').
3. Add edges to F from all places in ${}^*t$ to all transitions with the single initial place of the sub-net in their preset.
4. Add edges to F from all transitions in the sub-net with the termination place in their postset to all places in the post-set of t.
5. Add a control place c_t to the set of places, initialised with one token in the initial marking of the refined net, with outgoing edges from c_t to every transition with

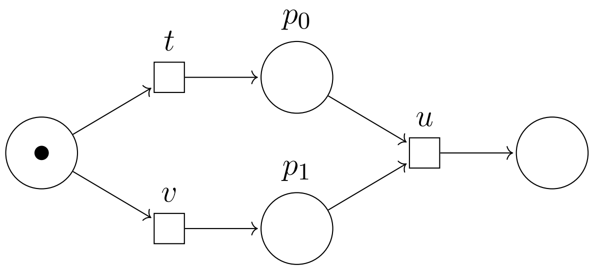


Figure 6.2

the single initial place of the sub-net in their preset, and incoming edges from every transition with the single termination place in their postset to c_t. This place ensures that at most one token is "inside" the new refined sub-net at a time, and prevents multiple tokens interacting in ways that were undefined on the original refinement specification net.

The labelling function ℓ' is augmented to return the appropriate label in the obvious way, for example for an unmodified transition t $\ell'(t) = \ell(t)$, and for a sub-net transition (t,t') $\ell'((t,t')) = \ell_{r(\ell(t))}(t')$ such that the correct label from the sub-net's labelling function is returned.

The **de-refinement** r^{-1} is a function that transforms the sub-nets produced by the refinement process back into the transitions they replaced, such that $r^{-1}(r(N)) = N$ where $=$ is algebraic equivalence. We also write $r^{-1}([M_r, \sigma_r])$ applied to ST-states to refer to the ST-state $[M_r, \sigma_r]$ lifted to the appropriate marking on the unrefined net, such that $[M, \sigma] = r^{-1}([M_r, \sigma_r])$ where:

1. The original marking M is the refined marking M_r restricted to the places of the original net. Any tokens within the internal places of the active sub-net are ignored.
2. The original firing sequence σ is the ordered sequences of transitions $t \in T$ that are currently in-progress. A transition t is considered in-progress if:
   - It has started: a sub-transition (t,t') has appeared in the refined firing sequence σ_r
   - It has not finished, its sub-net's designated termination place does not yet contain a token in the refined marking M_r

   The order of transitions in σ is inherited from the order in which their corresponding sub-nets were first activated in σ_r.

Now consider the trivial net with a single a transition, but with two tokens in p_0 in the initial marking, and the refinement net for a in Figure 6.2.

If we do not include the flow-control c_a places in the refined net, then two tokens could enter this refinement sub-net at a time and take different paths to introduce behaviour not exhibited by the original sub-net as it was defined. Nets such as the one in Figure 6.2 also violate the requirement that refinements do not contain any deadlock states, as with the single token both p_0 and p_1 are deadlock states.

6.3 EPST-bisimulation under Action Refinement

Theorem 6.1. EPST-bisimilarity is preserved under action refinement.

Proof. Given two nets N_0, N_1 over some set of visible actions A and a refinement r, then we need to show that $N_0 \leftrightarrow_{EPST} N_1 \implies r(N_0) \leftrightarrow_{EPST} r(N_1)$.

Let $\mathscr{R}$ be an EPST-bisimulation between N_0 and N_1, let us define an EPST-bisimulation $\mathscr{R}'$ between $r(N_0)$ and $r(N_1)$ as follows:

$$\begin{aligned}\mathscr{R}' :=\{([M_0', \sigma_0'], [M_1', \sigma_1'], R') \mid {} & \exists([M_0, \sigma_0], [M_1, \sigma_1], R) \in \mathscr{R} \text{ such that} \\ & r^{-1}([M_0', \sigma_0']) = [M_0, \sigma_0] \\ & \wedge\, r^{-1}([M_1', \sigma_1']) = [M_1, \sigma_1] \\ & \wedge\, R' := \{((t_0, v), (t_1, v)) \mid t_0 R t_1 \text{ and } v \textit{ is} \text{ an initial transition of the subnet as defined in Clause 3}\} \\ & \quad \cup \{((t_0, v), (t_1, v)) \mid \exists k.\sigma_0[k] = t_0 \wedge \sigma_1[k] = t_1 \\ & \qquad \text{and } v \textit{ is not} \text{ an initial transition of the subnet}\}\}\end{aligned}$$

Here $\mathrm{init}(\cdot)$ denotes the set of initial transitions of a sub-net, as defined in Clause 3.

We need to show that $\mathscr{R}'$ satisfies the EPST-bisimulation conditions:

1. Clause 1 requires that the initial states of the refined net are related. There exists some relation R such that $([M_0^{\text{in}}, \epsilon], [M_1^{\text{in}}, \epsilon], R) \in \mathscr{R}$, and the initial markings of the refined net de-refine to these ST-states, i.e. $r^{-1}([M_0^{\text{in}\prime}, \epsilon]) = [M_0^{\text{in}}, \epsilon]$ and $r^{-1}([M_1^{\text{in}\prime}, \epsilon]) = [M_1^{\text{in}}, \epsilon]$. Thus, there exists some R' constructed from R such that the initial states of the refined net are related in $\mathscr{R}'$.

2. Clause 2 requires that for $([M_0', \sigma_0'], [M_1', \sigma_1'], R')$, the transition R' forms a complete matching on the enabled transitions and preserves their concurrency structure. This follows by assumption that the transition relation R from the de-refined net relation $\mathscr{R}$ is a complete relation along with the construction of R'. The concurrency preservation property also holds as if two refined transitions originate from different original transitions, their concurrency depends only on the original transitions' presets, a property preserved by R. If they original from the same original transition, their concurrency depends on the internal structure of the refinement sub-nets, which follows as they are identical.

3. Clause 3 requires that for $([M_0', \sigma_0'], [M_1', \sigma_1'], R') \in \mathscr{R}'$ and transitions t_0' and t_1' such that $t_0' R' t_1'$, a number of properties hold: that their labels match; that the states produced by their firings are related by some transition relation R''; and that concurrency is preserved with respect to R' and R''. Let $t_0' = (t_0, v)$ and $t_1' = (t_1, v)$ for some unrefined transitions t_0 and t_1, there are two possibilities for the nature of t_0':

   (a) If v is an initial transition of the refinement net for t_0, in the sense of Clause 3, then t_0' firing corresponds to t_0 beginning to fire in the unrefined net. Let $[M_0, \sigma_0] = r^{-1}([M_0', \sigma_0'])$ and $[M_1, \sigma_1] = r^{-1}([M_1', \sigma_1'])$ be the de-refined ST-states that correspond to the current refined ST-state. As $t_0' R' t_1'$ we know that there exists some transition relation R such that $([M_0,\sigma_0],[M_1,\sigma_1],R) \in \mathscr{R}$ and by Clause 3 there must also exist a new relation R_{new} such that $([M_0 - {}^*t_0, \sigma_0 * t_0],[M_1 - {}^*t_1, \sigma_1 * t_1], R_{\text{new}}) \in \mathscr{R}$. Thus, as $r^{-1}([M_0' - {}^*t_0', \sigma_0' * t_0']) = [M_0 - {}^*t_0, \sigma_0 * t_0]$ and the same for $[M_1' - {}^*t_1, \sigma_1' * t_1']$, then there must exist a R_{new}' such that $([M_0' - {}^*t_0', \sigma_0' * t_0'],[M_1' - {}^*t_1', \sigma_1' * t_1'], R_{\text{new}}') \in \mathscr{R}'$. The properties of concurrency follow directly from the construction of R_{new}' and R'.

   (b) If v is an internal transition within the refinement net of t_0, then t_0' firing corresponds to no change in the unrefined net state. Let $[M_0,\sigma_0] = r^{-1}([M_0',\sigma_0'])$ and $[M_1,\sigma_1] = r^{-1}([M_1',\sigma_1'])$. Then $([M_0' - {}^*t_0', \sigma_0' * t_0'],[M_1' - {}^*t_1', \sigma_1' * t_1'], R') \in \mathscr{R}'$ for the same transition relation R', where concurrency is preserved trivially by the identity mapping.

4. Clause 4 follows from the symmetry of the above arguments and the symmetry of the underlying bisimulation.

5. Clause 5 requires that given $([M_0', \sigma_0' * t_0'\rho_0'],[M_1', \sigma_1' * t_1' * \rho_1'], R') \in \mathscr{R}'$ such that $|\sigma_0'| = |\sigma_1'|$, i.e. that t_0' and t_1' are at the same position k, that they can finish and the resulting state is also related in $\mathscr{R}'$, so there exists some R_{new}' such that $([M_0' + t_0'^*, \sigma_0' * \rho_0'],[M_1' + t_1'^*, \sigma_1' * \rho_1'], R_{\text{new}}') \in \mathscr{R}'$. There are two possibilities on the nature of $t_0' = (t_0, v)$ and $t_1' = (t_1, v)$ for unrefined transitions t_0, t_1 and refinement transition v:

   (a) If v is an internal action to the refinement net of t_0, then the state after t_0' finishes de-refines to the same ST-state $[M_0,\sigma_0]$, and therefore the resulting ST-states are also related in $\mathscr{R}'$.

   (b) If v is a terminating action to the refinement net of t_0, then the state after t_0' finishes de-refines to the ST-state equivalent to t_0 finishing. By assumption that $\mathscr{R}$ is an EPST-bisimulation, there exists some relation R_{new} such that

$$([M_0 + t_0^*, \sigma_0 * \rho_0],[M_1 + t_1^*, \sigma_1 * \rho_1], R_{\text{new}}) \in \mathscr{R},$$

   which implies that there exists a transition relation R_{new}' such that the resulting refined states are related in $\mathscr{R}'$, as required.

6. Clause 6 requires that the two nets agree on termination, that is for a triple $([M_0',\sigma_0'],[M_1',\sigma_1'],R') \in \mathscr{R}'$:

$$[M_0',\sigma_0'] = [\varnothing,\epsilon] \iff [M_1',\sigma_1'] = [\varnothing,\epsilon].$$

   If we assume that $[M_0',\sigma_0'] = [\varnothing,\epsilon]$, then by construction of $\mathscr{R}'$, $r^{-1}([M_0',\sigma_0']) = [\varnothing,\epsilon]$ implies that $[M_0,\sigma_0] = [\varnothing,\epsilon]$, by the same clause applied to $\mathscr{R}$ this implies that $[M_1,\sigma_1] = [\varnothing,\epsilon]$ and that therefore $r^{-1}([M_1',\sigma_1']) = [\varnothing,\epsilon]$. The only ST-state that de-refines to this is the terminal state, and therefore $r(N_1)$ is also in a terminal state. The same logic works symmetrically for the other direction of the implication, and therefore the two nets agree on termination.

Therefore, as $\mathscr{R}'$ is a valid EPST-bisimulation, EPST-bisimilarity is preserved under action refinement. $\square$

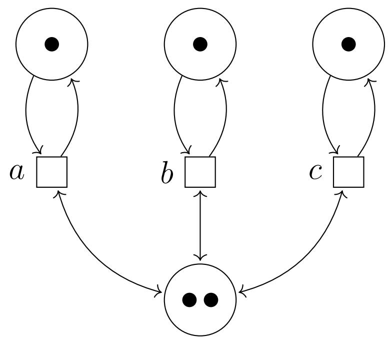
Chapter 7

Examples

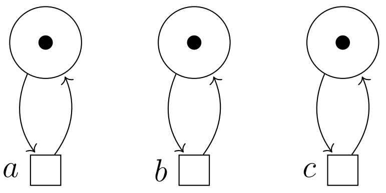
In this chapter we present some illustrative examples of nets that are considered equivalent by either Ep- or ST-bisimulation but not by EPST-bisimulation. Although these nets are conceptually simple, their purpose is to help the reader develop some intuition as to the practical discriminative power of EPST-bisimulation.

Example. Consider the two Petri nets in Figure 7.1. These two nets are Ep-bisimilar but not EPST-bisimilar. To see this, observe that each transition a, b, c is concurrent (in the sense of Definition 3.7) with each other in the sense that each firing does not impact the ability of the other two to fire, and therefore the two nets share the same concurrency structure. However, N_1 can enter an ST-state in which there are no tokens available and all transitions are in progress, for example $[\varnothing, (abc)]$, but N_0 can only have at most two transitions in progress at any one time as each transition requires a token from the bottom control place, and they are therefore not EPST-bisimilar.

Example. Consider the two Petri nets in Figure 7.2. These two nets are ST-bisimilar. The net N_1 is essentially two versions of N_0 stacked on top of each other with control places to ensure that we only ever take one side of the net at a time, and therefore from the perspective of ST-bisimulation any transition can be matched either in-progress or to fire. However, in N_1 the a and b transitions are not concurrent, whereas in N_0 they are, meaning that they have different concurrency structures and are therefore not EPST-bisimilar.

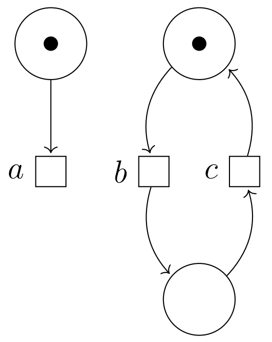


(a) N_0

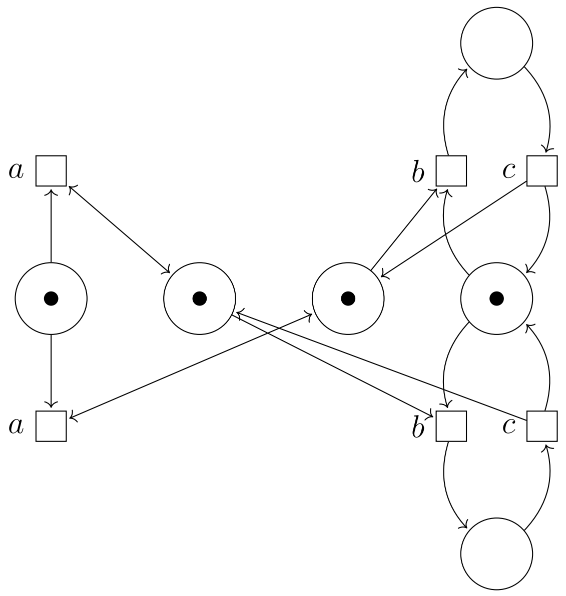


(b) N_1

Figure 7.1: Petri nets N_0 and N_1 for Example 7 illustrating the discrimination power of EPST-bisimulation versus Ep-bisimulation.



(a) N_0



(b) N_1

Figure 7.2: Two nets that are ST-bisimilar but not EPST-bisimilar.

Chapter 8

Conclusion

8.1 Summary of Contributions

This paper introduced and formalised the Enabling Preserving ST-bisimulation (EPST-bisimulation), a novel semantic equivalence for unsafe Petri nets. The primary motivation of this work was to fill a gap in the existing semantic equivalence lattice in the literature, which lacked a single relation that could simultaneously reason about systems with both real-time consistency and complex concurrency-dependent liveness properties. EPST-bisimulation achieves this by defining equivalence over states that track both in-progress firing sequences and a transition relation that rigorously preserves the concurrency structure of all enabled transitions.

We have rigorously established key theoretical properties of EPST-bisimulation. We proved that it forms a well-defined equivalence relation and crucially, that it is the coarsest semantic equivalence finer than both Ep- and ST-bisimulation. This result solidifies its natural position in the semantic equivalence lattice. Furthermore, to demonstrate its utility in compositional design, we proved that EPST-bisimulation is a congruence for the full algebra of ACP operators, and that it is preserved under action refinement. These key properties establish EPST-bisimulation as a powerful theoretical tool for the study of concurrent systems.

8.2 Evaluation and Limitations

Whilst this paper establishes a robust foundation for EPST-bisimulation, it is important to acknowledge and reflect on its trade-offs and limitations.

The primary trade-off we made was in its expressiveness versus its complexity. By incorporating detailed information about markings, firing sequences, and concurrency, the state space for EPST-bisimulation is significantly larger and more intricate than those of its predecessors. This detail is precisely what gives it its discriminating power, but it also makes the manual construction of bisimulations more challenging. The complexity of its construction highlights the utility of algorithmic approaches and automated tool support for applying EPST-bisimulation to large-scale practical problems.

Our work utilises Petri nets as its model of concurrency. Whilst we selected them for their ability to represent true concurrency natively, it's worth acknowledging alternative formalisms that could also have served this purpose. Prominent amongst these are *Event Structures*, which define concurrency through explicit causality and conflict relations between events [7].

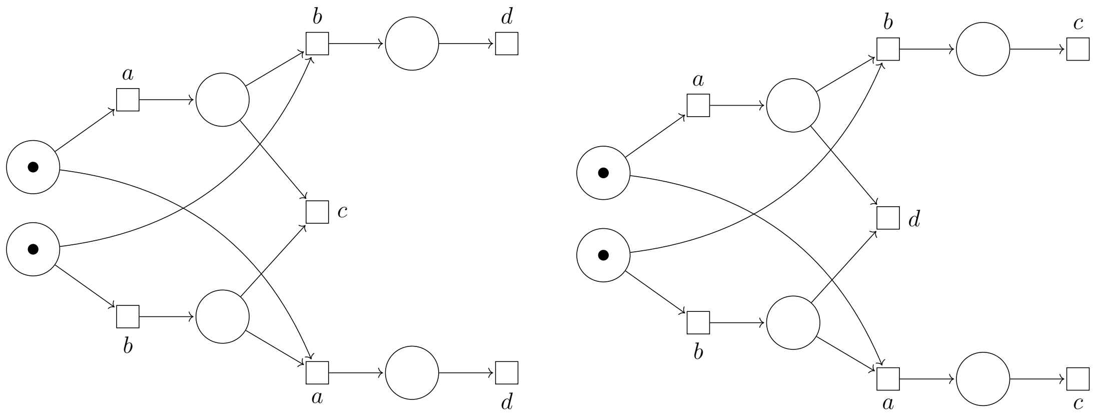
Finally, our definition of action refinement was restricted to well-behaved acyclic nets. This was necessary to guarantee the preservation of bisimulation properties in a straightforward manner. Investigation and extension of the framework to handle more general classes of refinements remains a non-trivial challenge likely to bear more theoretical fruit.

8.3 Future Work

This work opens up many avenues for future research. We suggest some potentially fruitful lines of inquiry that we would explore given additional time and resources.

- **Guarded Recursive Specifications**: A key direction for strengthening the algebraic theory is to prove that EPST-bisimulation works as an equivalence for *guarded recursive specifications*. A guarded recursive specification is a recursive specification in which every recursive process name is preceded by some non-recursive process a, for example the trivial example $X = a.X$. Proving this would involve showing that EPST-bisimulation can serve as the equivalence in such specifications, establishing that a definition like $X = a.X$ implies X.
- **Algorithmic Development and Tool Support**: A priority for making EPST-bisimulation a practical tool would be the development of a *decision algorithm*. That is, an algorithm that could decide, given two nets, whether an EPST-bisimulation exists between them. This would enhance its usefulness and allow the techniques described to be utilised in a model-checking context, enabling automated analysis of systems.
- **Weak and Branching Equivalents**: We defined EPST-bisimulation as a *strong* bisimulation, that is one that treats visible and internal τ actions identically. A natural extension would be to define weak and branching EPST-bisimulations that abstract away τ transitions to varying degrees. This would allow for the comparison of systems of varying levels of abstraction, a common practice in system design.

When translating the Ep-bisimulation to Petri nets from the original structure of LTSSs, an LTS-based structure augmented with a concurrency relation to capture which transitions were unaffected by others, we encountered an alternative, marginally less restrictive, formulation. In our initial definition, Clause 3 did not have the restriction that t_0 and t_1 were related by R, essentially meaning that the concurrency structures only had to match for a given marking and were *not* propagated throughout the entire net. This was a weakening of the original Ep-bisimulation definition, and meant that examples such as Figure 8.1 were equivalent under the original formulation by Glabbeek, Höfner, and Wang [2], but not under our definition. This presented a research question: *does this* *weaker formulation exhibit the same goodness properties of the original formulation?* Although an initial investigation indicated it would, we opted to strengthen our definition in order to utilise known results about the Ep-bisimulation. A full investigation of this weaker formulation is a promising direction for future research, as it could lead to a more general, minimally restrictive equivalence that still preserves justness.



(a) First net with c then d. (b) Second net with d then c.

Figure 8.1: Two Petri nets that are *not* Ep-bisimulation equivalent in the sense of Glabbeek, Höfner, and Wang [2], but are equivalent under our initial formulation.

Bibliography

[1] J.A. Bergstra and J.W. Klop. “Process algebra for synchronous communication”. In: *Information and Control* 60.1 (Jan. 1984), pp. 109–137. ISSN: 00199958. DOI: 10.1016/S0019-9958(84)80025-X.

[2] Rob van Glabbeek, Peter Höfner, and Weiyou Wang. “Enabling Preserving Bisimulation Equivalence”. In: (2012). DOI: 10.48550/arXiv.2108.00142.

[3] Rob J. van Glabbeek. “The Linear Time-Branching Time Spectrum (Extended Abstract)”. In: *Proceedings of the Theories of Concurrency: Unification and Extension*. CONCUR ’90. Berlin, Heidelberg: Springer-Verlag, Aug. 27, 1990, pp. 278–297. ISBN: 978-3-540-53048-0. DOI: 10.1007/bfb0039066.

[4] Rob Van Glabbeek and Peter Höfner. “Progress, Justness, and Fairness”. In: *ACM Computing Surveys* 52.4 (July 31, 2020), pp. 1–38. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3329125.

[5] Robert M. Keller. “Formal verification of parallel programs”. In: *Communications of the ACM* 19.7 (July 1976), pp. 371–384. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/360248.360251.

[6] T. Murata. “Petri nets: Properties, analysis and applications”. In: *Proceedings of the IEEE* 77.4 (Apr. 1989), pp. 541–580. ISSN: 00189219. DOI: 10.1109/5.24143.

[7] Mogens Nielsen, Gordon Plotkin, and Glynn Winskel. “Petri nets, event structures and domains, part I”. In: *Theoretical Computer Science* 13.1 (1981), pp. 85–108. ISSN: 03043975. DOI: 10.1016/0304-3975(81)90112-2.

[8] “On the relationship of CCS and petri nets”. In: Ursula Goltz and Alan Mycroft. *Lecture Notes in Computer Science*. ISSN: 0302-9743, 1611-3349. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984, pp. 196–208. ISBN: 978-3-540-13345-2 978-3-540-38886-9. DOI: 10.1007/3-540-13345-3_18.

[9] David Park. “Concurrency and automata on infinite sequences”. In: *Theoretical Computer Science*. Ed. by Peter Deussen. Vol. 104. Series Title: Lecture Notes in Computer Science. Berlin/Heidelberg: Springer-Verlag, 1981, pp. 167–183. ISBN: 978-3-540-10576-3. DOI: 10.1007/BFb0017309.

[10] Carl Adam Petri. “Kommunikation mit Automaten”. PhD thesis. Universiät Hamburg, June 20, 1962. URL: https://edoc.sub.uni-hamburg.de/informatik/volltexte/2011/160/ (visited on 07/30/2025).

[11] Rob Van Glabbeek and Frits Vaandrager. “Petri net models for algebraic theories of concurrency: extended abstract”. In: *PARLE Parallel Architectures and Languages Europe*. Ed. by J. W. De Bakker, A. J. Nijman, and P. C. Treleaven. Red. by G. Goos et al. Vol. 259. Series Title: Lecture Notes in Computer Sci-

ence. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 224–242. ISBN: 978-3-540-17945-0 978-3-540-47181-3. DOI: `10.1007/3-540-17945-3_13`.

Appendix A

Semantic Equivalence Lattice

As discussed the *semantic equivalence lattice* is a device to visualise the relationship between various semantic equivalences. It is essentially a directed graph on which nodes are semantic equivalences and directed edges indicate that the source is finer than the destination. They are typically organised along some additional axes to indicate the nature of the equivalence, we present a version in Figure A.1 organised along a horizontal axis indicating whether they abstract from or include concurrency information, and a vertical axis indicating linear time versus branching time.

Linear time semantic equivalences are only concerned with the visible actions of a system, and are not concerned with the internal structure of the processes that exhibit the visible behaviour. For example, strong complete trace semantics states that two equivalences are equivalent if they have the same complete traces, such that the processes $a||b$ and $a.b+b.a$ are equivalent as they both have completed traces ab and ba.

$$\begin{tikzcd}
\approx_{tree} \arrow[dd] & & & & & \equiv_{occ} \arrow[lllll] \arrow[d] \\
 & \underline{\leftrightarrow}_{Ep} \arrow[dl] & & & \underline{\leftrightarrow}_{EPST} \arrow[lll] \arrow[dl] & \underline{\leftrightarrow}_{sp} \arrow[l] \arrow[dl] \\
\approx_{ib} \arrow[d] & \approx_{sb} \arrow[l] \arrow[d] & \approx_{2b} \arrow[l] \arrow[d] & \underline{\leftrightarrow}_{ST} \arrow[l] \arrow[d] & \approx_{h} \arrow[l] \arrow[d] & \\
\approx_{it} & \approx_{st} \arrow[l] & \approx_{2t} \arrow[l] & \approx_{STt} \arrow[l] & \approx_{pt} \arrow[l] &
\end{tikzcd}$$

Figure A.1: A spectrum of semantic equivalences on Petri nets

Appendix B

Notation Table

Table B.1: Table of Notation

Symbol	Description
General Mathematical Notation	
A	A set of actions.
τ	The internal, unobservable action.
ϵ	Successful termination.
δ	Deadlock or failure.
$\mathbb{N}$	The set of natural numbers (including 0).
$\mathscr{P}(X)$	The powerset of a set X.
X^*	The set of all finite sequences (strings) over a set X.
Petri Net Notation	
N	A Petri net, defined as the tuple $(S,T,F,M^{\text{in}},\ell)$.
S	The set of places in a net.
T	The set of transitions in a net.
F	The flow relation, $F \subseteq (S \times T) \cup (T \times S)$.
M	A marking (configuration of tokens) of a net.
M^{in}	The initial marking of a net.
ℓ	The labelling function, $\ell : T \to A$.
${}^*t$	The pre-set of a transition t; the set of its input places.
t^*	The post-set of a transition t; the set of its output places.
$\text{en}(M)$	The set of transitions enabled in marking M.
$M[t\rangle M'$	From marking M, transition t fires, resulting in marking M'.
ST-Bisimulation Notation	
$[M,\sigma]$	An ST-state, consisting of a marking M and a firing sequence σ.
$\sigma \in T^*$	A sequence of transitions currently in the process of firing.
t^+	An event indicating that transition t has started firing.
t^{-k}	An event indicating the k^{th} in-progress instance of t has finished.
π	A path or an ST-path in a net.

Continued on next page

Table B.1 – continued from previous page

Symbol	Description
Ep- and EPST-Bisimulation Notation	
$\smile\!\!\bullet$	Asymmetric concurrency relation ($t \smile\!\!\bullet u$ means u firing does not disable t).
$\smile$	Symmetric concurrency relation ($t \smile u \iff t \smile\!\!\bullet u \wedge u \smile\!\!\bullet t$).
$\mathscr{R}$	A bisimulation relation; a set of related state tuples.
R	A relation on transitions, $R \subseteq T_0 \times T_1$, used within an Ep- or EPST-bisimulation tuple.
Process Algebra (ACP) Notation	
$P.Q$	Sequential composition.
$P+Q$	Choice.
$P\|\|Q$	Parallel composition.
γ	A communication function, $\gamma : A \times A \to A$.
$\partial_H(P)$	Encapsulation (restriction) of actions in set H.
$\tau_I(P)$	Abstraction of actions in set I to τ.