Master's Thesis

A Transformation of Hybrid Petri Nets with General Firings to Stochastic Hybrid Automata

by
Lena Verscht

1st Examiner:
Prof. Dr. Erika Ábrahám

2nd Examiner:
apl. Prof. Dr. Thomas Noll

The present work was submitted to:
LuFG Theory of Hybrid Systems
RWTH Aachen University

November 21, 2022

Contents

1. Introduction

Our everyday life depends on many complex critical infrastructures, ranging from water treatment facilities to power plants. At the same time, technological advances cause more and more areas to depend on automated systems, as can be observed, for example, in smart cars and smart homes. Securing and analyzing such systems is essential to avoid failures and prevent serious repercussions, making it an important area of research.

In computer science, systems are often assumed to be discrete. This is reasonable insofar as program executions can be viewed as a series of discrete steps. However, real-world systems are often dynamic and depend on both discrete and continuous variables. A simple example of a hybrid system is the heating system in a smart home, which reacts to the continuously changing temperature and the discrete settings of the thermostat. *Hybrid* modeling mechanisms were developed to properly describe the behavior of such systems.

Two widespread approaches for modeling hybrid systems are hybrid automata and hybrid Petri nets. Hybrid automata extend discrete automata by adding specifications for the continuous evolution of variables to each location, implicitly adding the concept of time. Similarly, a Petri net can also be seen as the extension of a discrete automaton, allowing a location, here referred to as a place, to hold several so-called tokens. Places are connected by transitions that move tokens between them. A hybrid Petri net broadens the modeling power of Petri nets essentially by adding continuous places holding fluid instead of discrete tokens and corresponding continuous transitions.

In many cases, real-world systems are under the influence of an uncertain environment. For example, components of the system might fail with a certain probability or depend on external factors such as severe weather, whose occurrence can be characterized by a probability distribution. This makes *probabilistic* hybrid systems a cornerstone for the analysis of safety-critical infrastructures.

For both modeling approaches mentioned above, several stochastic variants exist. In a recent work, Gerlach presented a compositional modeling language for *stochastic hybrid automata* (SHAs), which will be used in this thesis. Gribaudo and Remke introduced *hybrid Petri nets with general firings* (HPnGs), which essentially add stochasticity to hybrid Petri nets by delaying the firing of transitions with respect to a probability distribution [GR16]. HPnGs were proven to be useful for modeling real-world systems.

For example, Ghasemieh et al. studied a waste water treatment facility in Enschede, the Netherlands [GRH16].

There exist various techniques and frameworks for analyzing SHAs, for example, reachability analysis, which handles the problem of computing the reachability probability for a given set of states [Aba+07; Frä+11; Hah+13; HH14]. Similar results would be desirable for HPnGs, which can be achieved in two ways: Either by examining methods of interest and transfer them to the setting of HPnGs [PR17; HR19; Hül+21], or by providing a method to transform HPnGs into SHAs [Pil+20]. The latter has the strong advantage that all methods are directly applicable without the need to consider them separately. In this thesis, we define such a transformation and prove its correctness. A first approach to this was made by Pilch et al. in [Pil+20] by simulating time-bounded behavior of HPnGs with a subclass of SHAs. The restriction to paths of a bounded length is a consequence of employing a symbolic calculation tree as an intermediate step of the transformation. We adopt a different approach without the need for symbolic computations. As an advantage, the transformation proposed in this thesis is not limited by a time bound and instead can accurately model the behavior of a HPnG for infinite paths.

There are some fundamental differences between SHAs and HPnGs. For example, time can stop when certain conditions are violated in SHAs. In HPnGs, time always passes and the components must always adapt their behavior such that no required condition is violated. We will address and resolve these challenges. The main contributions of this thesis are summarized in the following:

- We propose a novel algorithm for solving conflicts concerning the under- and overflow of places within the continuous part of the HPnG, referred to as the *rate adaption* and prove its termination on a subclass of HPnGs.
- We give a rigorous definition of the operational semantics of HPnGs including the rate adaption by giving a set of inference rules, building on the syntax and semantics of HPnGs from previous publications [GR16; Pil+20].
- We define a formal transformation of HPnGs to semantically equivalent SHAs for unbounded executions.
- We prove the correctness of the transformation by defining a bisimulation between the states of both systems.

Structure. The objective of this thesis is to provide a method for transforming HPnGs into SHAs. To this end, we provide a rigorous definition of both modeling structures, starting with SHAs in Chapter 2. We present HPnGs in Chapter 3 and their syntax in Section 3.1. To describe their semantics, we extend existing definitions towards a

more precise and formal description through a set of inference rules in Section 3.2. This includes an algorithm for the rate adaption in Section 3.2.2.

In Chapter 4, we formally define a compositional transformation of HPnGs to SHAs, including a detailed discussion on possible variants and approaches. The transformation is split into a transformation of the discrete fragments in Section 4.2, a transformation of continuous places in Section 4.3 and finally the composition of the separate parts in Section 4.4. We prove the correctness of the transformation in Section 4.5.

Finally, we draw a conclusion in Chapter 5. In Appendix A, we give an overview of used notations and provide some omitted proofs in Appendix B.

2. Stochastic Hybrid Automata

Discrete systems evolve by moving from one state to another, taking discrete steps. Automata or transition systems can be used to model such systems, including for example program executions, which is why they are widely used in computer science. Hybrid automata extend these models by adding the concept of time. In addition to jumping from one location to another, we can let time pass while staying in a location and let variables evolve continuously as specified by differential equations. This makes hybrid automata useful for modeling dynamic systems that exhibit both discrete and continuous behavior, such as heating systems or the charging process of a battery.

Real-world models are often exposed to some kind of uncertainty. Stochastic hybrid automata (SHAs) were introduced to capture this in a meaningful way. They extend hybrid automata with random components, such as sampling when to take a jump from a specified distribution.

There are several definitions of SHAs that differ, for example, in the precise deployment of stochasticity [Pol+03; JP09; Ger22]. We choose to build the transformation on a compositional model presented in a very recent Master's thesis by Gerlach [Ger22]. For the transformation, compositionality has the advantage that it allows us to split the Petri net into smaller parts, transform those separately, and compose the result. Therefore, we obtain simpler definitions compared to non-compositional approaches.

We begin by defining the syntax of SHAs in Section 2.1 and their semantics in Section 2.2. Since we aim to define a compositional transformation, we define the composition of SHAs in Section 2.3. Some minor details differ from Gerlach's approach, which will be discussed in more detail in Section 2.4.

2.1. Syntax of SHAs

SHAs manipulate a set of variables Var, partitioned into disjoint sets of controlled and non-controlled variables. The latter allow us to account for factors that the SHA cannot influence but that potentially influence the SHA. The assignment of values to the variables is described by a function referred to as *valuation* v.

Definition 2.1. Valuations
A *valuation* over a set of variables *Var* is a function $\nu : Var \to \mathbb{R}$. We denote the set of valuations over *Var* as $\mathcal{V}_{Var}$. If *Var* is clear from context, we omit the index.

In order to manipulate a valuation $\nu \in \mathcal{V}_{Var}$, we define $\nu[x \mapsto a] \in \mathcal{V}_{Var}$ as the valuation where the value of variable $x \in Var$ is replaced by $a \in \mathbb{R}$, i.e., for all $y \in Var$:

$$\nu[x \mapsto a](y) = \begin{cases} a & \text{if } y = x \\ \nu(y) & \text{else.} \end{cases}$$

For the sake of readability, we write $\nu[x \mapsto y]$ for $\nu[x \mapsto \nu(y)]$ when we want to set the value of x to the value of $y \in Var$. Also, we write $\nu[x \mathrel{+}= 1]$ for $\nu[x \mapsto \nu(x) + 1]$ and dually $\nu[x \mathrel{-}= 1]$ for $\nu[x \mapsto \nu(x) - 1]$.

As mentioned earlier, the variables of SHAs evolve continuously over time. Their precise evolution is described by a set of *activities*. Later, this set will often be characterized by differential equations.

Definition 2.2. Activities
An *activity* is a continuous function $f : \mathbb{R}_{\geq 0} \to \mathcal{V}_{Var}$. We denote the set of activities with respect to *Var* as $\mathcal{F}_{Var}$. If *Var* is clear from context, we omit the index.

We distinguish between variables that are controlled by the SHA and those that are not. For this, we formally define what it means for a set of valuations to be independent of a variable set.

Definition 2.3. Closure of Sets
Let *Con* and *NCon* be two disjoint variable sets and let $Var = Con \dot\cup NCon$. For all $\nu, \nu' \in \mathcal{V}_{Var}$, we define $\nu \approx_{Con} \nu'$ iff ν and ν' agree on the values for all variables in *Con*, i.e., it holds that $\nu(v) = \nu'(v)$ for all $v \in Con$. Analogously, we define $\nu \approx_{NCon} \nu'$ if $\nu(v) = \nu'(v)$ for all $v \in NCon$. Then, we call a set $S \subseteq \mathcal{V}_{Var}$ of valuations *NCon-closed* iff $S = \{\nu' \in \mathcal{V}_{Var} \mid \exists \nu \in S.\ \nu \approx_{Con} \nu'\}$.

Thus, S is closed with respect to a set of non-controlled variables *NCon* if it only specifies values for variables from *Con* and allows for all possible valuations of the variables in *NCon*.

The probabilistic components in the SHA follow *probability distributions*. We denote the set of all distributions over the non-negative reals as *Distr*, so

$$Distr = \left\{ p : \mathbb{R}_{\geq 0} \to [0,1] \;\middle|\; \int_{x \in \mathbb{R}_{\geq 0}} p(x)\, dx = 1 \right\}.$$

The *support* of a probability distribution $p \in Distr$ is denoted by $supp(p)$, i.e.,

$$supp(p) = \{x \in \mathbb{R}_{\geq 0} \mid p(x) > 0\}.$$

A distribution p is *continuous* if $supp(p)$ is uncountable.

Next, we define some distributions that will be used in this thesis. For $a, b \in \mathbb{R}_{\geq 0}$ with $a \leq b$, we denote by $\mathcal{U}_{[a,b]} \in Distr$ the uniform distribution over the interval $[a, b] \subset \mathbb{R}$. More precisely, for all $x \in \mathbb{R}_{\geq 0}$

$$\mathcal{U}_{[a,b]}(x) = \begin{cases} \frac{1}{b-a} & \text{if } x \in [a,b] \\ 0 & \text{else.} \end{cases}$$

Another common distribution is the *exponential distribution* $exp \in Distr$ given by

$$exp(x) = e^{-x}$$

for all $x \in \mathbb{R}_{\geq 0}$. We will also need probability distributions that assign all probability mass to one single value. For this purpose, we define the *Dirac distribution* $\delta_a \in Distr$ for $a \in \mathbb{R}_{\geq 0}$ for all $x \in \mathbb{R}_{\geq 0}$ as

$$\delta_a(x) = \begin{cases} 1 & \text{if } x = a \\ 0 & \text{else.} \end{cases}$$

For a random variable $a \in Var$ and a probability distribution $p \in Distr$ we write $a \sim p$ to express that a is distributed according to p. Now, we can formally define the syntax of SHA.

DEFINITION 2.4. **Syntax of Stochastic Hybrid Automata**

A *stochastic hybrid automaton* (SHA) is a tuple

$$\mathcal{A} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

with the following components:

- Loc is a non-empty finite set of *locations*.
- $Var = Con \uplus NCon$ is a finite set of *variables*, partitioned into disjoint sets of controlled and non-controlled variables.
- $Inv\colon Loc \to 2^{\mathcal{V}}$ assigns an *invariant* to each location $l \in Loc$ such that $Inv(l)$ is $NCon$-closed.
- $Init\colon Loc \to 2^{\mathcal{V}}$ assigns a set of *initial valuations* to each location $l \in Loc$ such that $Init(l)$ is $NCon$-closed and $Init(l) \subseteq Inv(l)$.
- $Edge \subseteq \mathbb{N} \times Loc \times 2^{\mathcal{V}^2} \times Loc$ is a finite set of *jumps* such that for all $(id, l, \mu, l') \in$

Edge and all $(\nu, \nu') \in \mu$ it holds that $\nu \approx_{NCon} \nu'$ and $\nu' \in Inv(l')$. We call l the *source location* and l' the *target location* of a jump $(id, l, \mu, l') \in Edge$.

- $Act\colon Loc \to 2^{\mathcal{F}}$ assigns a set of *activities* to each location.
- *Lab* is a set of *labels* specifying some random processes.
- $Proc\colon Edge \to Lab$ assigns a random process to each jump such that all $e_1, e_2 \in Edge$ with the same source location have $Proc(e_1) \neq Proc(e_2)$.
- $Dur\colon Lab \times Loc \times \mathcal{V} \to Distr$ assigns a probability distribution to each label, location and valuation in $Lab \times Loc \times \mathcal{V}$.
- $Wgt\colon Edge \to \mathbb{N}_{>0}$ assigns a positive weight to each jump.

Locations. What is commonly called a state in the context of discrete automata is now referred to as a location. Then, a state of an SHA additionally includes a valuation of the variables.

Variables. SHAs manipulate a set of variables, which are split into controlled and non-controlled variables. Controlled variables are those that the SHA can read and write, while non-controlled are for reading only.

Invariants. In each location, we can place conditions on the variables. An SHA cannot remain in a location if the invariant is violated, meaning that the current valuation of the variables must be contained in the set of valuations specified by the invariant of the current location at all times.

We require the invariant to be closed with respect to the set *NCon* of non-controlled variables to ensure that the SHA does not implicitly restrict the environment's behavior. If the invariant would depend on a non-controlled variable, it might happen that the environment attempts to reset this variable, but is prevented by the invariant which must not be violated. Thus, we must require that the invariant is *NCon*-closed.

Initial Valuations. As for invariants, the initial valuations are given as sets of valuations for each location l, specifying which valuations are allowed when starting in l. If $Init(l)$ is empty, an execution of the SHA cannot start in location l. For similar reasons as above, we require the set of initial valuations to be *NCon*-closed. Additionally, each initial valuation must satisfy the invariant.

Jumps. The jumps specify how to move between locations. A jump (id, l, μ, l') starts in the source location l and leads to the target location l'. The set μ of valuation pairs places conditions on the valuations before and after taking the jump. We refer to the conditions $\{\nu \in \mathcal{V} \mid \exists \nu' \in \mathcal{V}.\ (\nu, \nu') \in \mu\}$ on the starting valuation as *guards* and to the conditions $\{\nu' \in \mathcal{V} \mid \exists \nu \in \mathcal{V}.\ (\nu, \nu') \in \mu\}$ on the target valuation as *effect*. Effects must not change non-controlled variables. However, the guard may depend on the non-controlled variables. A jump can only be taken if there exists $(\nu, \nu') \in \mu$ with ν being the current valuation and ν' satisfying the invariant of the target location l'. The jump identifier *id* is added for technical reasons for the definition of the composition of SHAs. To improve readability, it is often omitted.

Activities. For every location $l \in Loc$, the set of activities describes how the variables are evolving when the SHA is in location l. We specify $Act(l)$ by a first-order system of linear ordinary differential equations of the form $\dot{x}_{Con} = A_l x_{Con} + B_l x_{NCon} + c_l$ where

- x_{Con} and x_{NCon} are vectors of the controlled respectively non-controlled variables, and $\dot{x}_{Con}$ represents the first derivative of the evolution of the controlled variables,
- A_l and B_l are real-valued matrices of suitable dimensions, and
- c_l is a vector of real-valued constants.

$Act(l)$ then consists of all activities satisfying these equations.

Labels. The set of labels is used to relate jumps that model the same random process.

Random Processes. Every jump is assigned a label corresponding to a random process. A label can be assigned to several jumps as long as the jumps are not originating in the same location. This is required to prevent random processes from being in a race with themselves.

Duration. A jump is not necessarily taken as soon as possible. Instead, the function *Dur* assigns a probability distribution to each label, depending on the current valuation and location. Then, the jump is taken when its random process was enabled for the number of time units sampled from the distribution assigned to the corresponding label. For this reason, $Dur(a, l, \nu)$ is often referred to as the *enabling duration* of the jump e with $Proc(e) = a$.

Weight. Each jump is assigned a positive weight, which will be used to solve conflicts between jumps that can be taken at the same time.

In the remaining thesis, we let

- $\mathcal{A} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt)$,
- $\mathcal{A}_1 = (Loc_1, Var_1, Inv_1, Init_1, Edge_1, Act_1, Lab_1, Proc_1, Dur_1, Wgt_1)$, and
- $\mathcal{A}_2 = (Loc_2, Var_2, Inv_2, Init_2, Edge_2, Act_2, Lab_2, Proc_2, Dur_2, Wgt_2)$.

The first six components of an SHA, namely *Loc*, *Var*, *Inv*, *Init*, *Edge*, and *Act*, form a hybrid automaton. We refer to this as the underlying hybrid automaton. Thus, SHAs extend hybrid automata by the four components *Lab*, *Proc*, *Dur*, and *Wgt*. Stochasticity is introduced at two points: By the function *Dur* for deciding the enabling duration of jumps and by the weights *Wgt*, which are used to resolve conflicts probabilistically.

Still, non-determinism might arise in the model. For example, the initial state is not necessarily unique. While part of the non-determinism is solved using probabilities, we forbid all other sources of non-determinism.

DEFINITION 2.5. **Deterministic Stochastic Hybrid Automata**

Let $\mathcal{A}$ be an SHA. We say that $\mathcal{A}$ is *deterministic* (with respect to initial valuations, resets, and activities) if the following conditions hold:

- There exists $l \in Loc$ such that $Init(l) \neq \emptyset$, $Init(l') = \emptyset$ for all $l' \neq l$ and $\nu \approx_{Con} \nu'$ for all $\nu, \nu' \in Init(l)$.
- For each $(id, l, \mu, l') \in Edge$ and $\nu \in \mathcal{V}$ there is at most one $\nu' \in \mathcal{V}$ such that $(\nu, \nu') \in \mu$.
- For each $l \in Loc$, $\nu \in \mathcal{V}$, and evolution of the environment $f_N \in \mathcal{F}_{NCon}$ with $f_N(0) = \nu|_{NCon}$, there exists a unique $f \in Act$ such that $f \approx_{NCon} f_N$ and $f(0) = \nu$.

In the following, we always assume the SHAs to be deterministic. Intuitively, each SHA then has a unique starting location l and a unique initial valuation of the controlled variables. For every jump, the effect is unique and only depends on the current valuation. The same holds for the activities, i.e., when we are in a location l, the evolution of the variables is fixed by the current valuation and the evolution of the non-controlled variables.

EXAMPLE 2.1. **Syntax of SHA**

Consider the SHA $\mathcal{A}_1$ depicted in Figure 2.1. The two locations l_0 and l_1 are controlling a variable x. For later examples, we assume there exists a non-controlled variable y that does not affect $\mathcal{A}_1$. The initial location is l_0 and requires x to be zero in the initial valuation. While $\mathcal{A}_1$ is in l_0, x is increased by two per time unit and is not allowed to exceed eight. The jump to l_1 can be taken as soon as x is at least six,

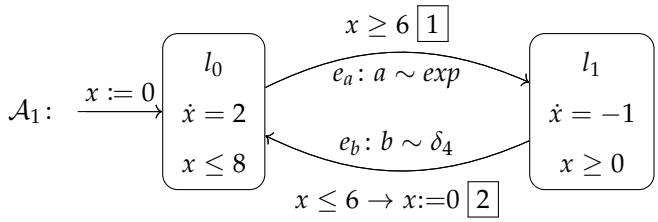


Figure 2.1.: Depiction of the SHA $\mathcal{A}_1$ discussed in Example 2.1. The name of each jump is denoted with the label and the assigned probability distribution. Guards and effects are denoted next to the corresponding jumps as *guard* $\to$ *effect*. Weights are denoted in boxes.

and the jump is taken after being enabled for t time units, where the value for t is sampled from the exponential distribution *exp*.

In l_1, x is decreased by one and must remain non-negative. The jump to l_0 is assigned a Dirac distribution, meaning that the jump will be taken after being enabled for four time units. The condition for being enabled is $x \leq 6$, and x is reset to zero when the jump is taken.

Formally, we have

$$\mathcal{A}_1 = (Loc_1, Var_1, Inv_1, Init_1, Edge_1, Act_1, Lab_1, Proc_1, Dur_1, Wgt_1)\,,$$

with the following components:

- $Loc_1 = \{l_0, l_1\}$.
- $Var_1 = \{x, y\}$ with $Con_1 = \{x\}$ and $NCon_1 = \{y\}$.
- $Inv_1(l_0) = \{\nu \in \mathcal{V} \mid \nu(x) \leq 8\}$ and $Inv_1(l_1) = \{\nu \in \mathcal{V} \mid \nu(x) \geq 0\}$.
- $Init_1(l_0) = \{\nu \in \mathcal{V} \mid \nu(x) = 0\}$ and $Init_1(l_1) = \emptyset$.
- $Edge_1 = \{e_a, e_b\}$ with
  - $e_a = (l_0, \mu_a, l_1)$, $\mu_a = \{(\nu, \nu') \mid \nu(x) \geq 6 \wedge \nu(y) = \nu'(y)\}$ and
  - $e_b = (l_1, \mu_b, l_0)$, $\mu_b = \{(\nu, \nu') \mid \nu(x) \leq 6 \wedge \nu'(x) = 0 \wedge \nu(y) = \nu'(y)\}$.
- $Act_1(l_0)$ is the set of activities that are solutions of $\dot{x} = 2$ and $Act_1(l_1)$ is the solution set of $\dot{x} = -1$.
- $Lab_1 = \{a, b\}$.
- $Proc_1(e_a) = a$ and $Proc_1(e_b) = b$.
- $Dur_1(a, (l_0, \nu)) = exp$ and $Dur_1(b, (l_1, \nu)) = \delta_4$ for all $\nu \in \mathcal{V}$.
- $Wgt_1(e_a) = 1, Wgt_1(e_b) = 2$.

Note that we omit the identities from the jumps and implicitly assign two unique values. The weight that is assigned to the jumps in this example does not influence the behavior, because the jumps are never in conflict.

2.2. Semantics of SHAs

The semantics of an SHA describes its behavior by specifying how one state can be reached from another. For this, we first define what a state of an SHA consists of.

Definition 2.6. State of an SHA
Let $\mathcal{A}$ be an SHA. Define $V_{Lab} = \{c_a \mid a \in Lab\}$ and let $\mathcal{V}_{Lab}$ be the set of valuations over V_{Lab}. Then, the set of *states* of $\mathcal{A}$ is

$$\Theta^{\mathcal{A}} = Loc \times \mathcal{V} \times \mathcal{V}_{Lab}.$$

A state (l, ν, ν_{Lab}) is *initial* if $\nu \in Init(l)$ and ν_{Lab} is sampled from the corresponding distributions, meaning for $a \in Lab$, we sample the value for c_a from $Dur(loc, \nu, r)$.

In addition to the current location and the current valuation, a state (l, ν, ν_{Lab}) therefore additionally contains a valuation over the set of labels, or formally over clock variables for each label. These variables will be initialized with the enabling durations sampled from the probability distributions assigned to the labels, and then counted down to zero as long as a corresponding jump is enabled. For simplicity, we write $\nu_{Lab}(a)$ instead of $\nu_{Lab}(c_a)$ for $a \in Lab$.

Next, we define what it means for a jump to be enabled. Intuitively, this is the case if we are in the starting location of the jump and the conditions given by the guard of the jump are fulfilled.

Definition 2.7. Enabled Jumps
Let $\mathcal{A}$ be an SHA. For a jump $e = (id, l, \mu, l') \in Edge$ and a state $\vartheta = (l, \nu, \nu_{Lab})$ of $\mathcal{A}$, we define $enabled_{\vartheta}(e)$ to be true iff ν satisfies the guard of the jump, i.e. we have $(\nu, \nu') \in \mu$ for some $\nu' \in \mathcal{V}$.

Note that technically, $enabled_{\vartheta}(e)$ only depends on the valuation ν and not on the full state ϑ. For consistency, we nevertheless use ϑ in the definition.

When taking a jump e starting in a valuation ν, the reached valuation ν' is unique, as we assume the SHA to be deterministic. Since we will need to reference this valuation, we define another predicate that identifies ν'.

Definition 2.8. Discrete Step

Let $\mathcal{A}$ be an SHA. For a jump $e = (id, l, \mu, l') \in Edge$ and a state $\vartheta = (l, \nu, \nu_{Lab})$ of $\mathcal{A}$ such that $enabled_\vartheta(e)$ holds, we define $disc_\vartheta(e) \in \mathcal{V}$ to be the unique valuation reached after taking jump e from ϑ, i.e., $disc_\vartheta(e) = \nu'$ for the unique $(\nu, \nu') \in \mu$.

A jump is only taken if it is enabled and additionally, the corresponding clocks are zero. This is expressed in the next definition.

Definition 2.9. Fireable Jumps

Let $\mathcal{A}$ be an SHA. For a jump $e = (id, l, \mu, l') \in Edge$ and a state $\vartheta = (l'', \nu, \nu_{Lab})$ of $\mathcal{A}$, we define $fireable_\vartheta(e)$ to be true iff $l = l''$, $enabled_\vartheta(e)$ and $\nu_{Lab}(Proc(e)) = 0$. We then say that e is *fireable* in ϑ.

The predicates defined above describe conditions on the discrete behavior of an SHA. For the continuous part of the SHA, we also define two predicates: One that expresses how the activities control the evolution of variables, and one that deals with the correct evolution of clock variables.

Definition 2.10. Time Step

Let $\mathcal{A}$ be an SHA. For a state $\vartheta = (l, \nu, \nu_{Lab})$ of $\mathcal{A}$, $f \in Act(l)$ and $\tau \in \mathbb{R}_{>0}$, we define $time^f_\vartheta(\tau) = \nu'$ iff

- $f(0) = \nu$,
- $f(\tau) = \nu'$, and
- $f(\tau') \in Inv(l)$ for all $\tau' \in [0, \tau]$.

Otherwise, $time^f_\vartheta(\tau)$ is undefined.

Thus, we have $time^f_\vartheta(\tau) = \nu'$ if, starting from state ϑ, we reach the valuation ν' after τ time units by application of activity f. Note that ν' is uniquely determined by f and ϑ for every evolution of the non-controlled variables, as we assume $\mathcal{A}$ to be deterministic.

When time passes, we do not only have to update the valuation of the variables but also the valuation of the clocks. This is expressed in the next predicate.

Definition 2.11. Evolution of Clocks

Let $\mathcal{A}$ be an SHA. For a state $\vartheta = (l, \nu, \nu_{Lab}) \in \Theta^{\mathcal{A}}$, $\nu'_{Lab} \in \mathcal{V}_{Lab}$, $\tau \in \mathbb{R}_{>0}$ and $f \in Act(l)$ such that $f(0) = \nu$, we define $clocks^f_\vartheta(\tau) = \nu'_{Lab}$ iff for all $r \in Lab$, one of the following is true:

- There exists some $e = (id, l, \mu, l') \in Edge$ with $r = Proc(e)$ such that

1. $\forall \tau' \in (0,\tau).\ enabled_{\vartheta'}(e)$ for $\nu' = time^f_\vartheta(\tau')$ and $\vartheta' = (l,\nu',\nu_{Lab})$, and
2. $\nu'_{Lab}(r) = \nu_{Lab}(r) - \tau \geq 0$,

- or for all $e = (id,l,\mu,l') \in Edge$ with $r = Proc(e)$ it holds that
  1. $\neg\exists \tau' \in (0,\tau).\ enabled_{\vartheta'}(e)$ for $\nu' = time^f_\vartheta(\tau')$ and $\vartheta' = (l,\nu',\nu_{Lab})$, and
  2. $\nu'_{Lab}(r) = \nu_{Lab}(r)$.

Otherwise, $clocks^f_\vartheta(\tau)$ is undefined.

Intuitively, if $clocks^f_\vartheta(\tau) = \nu'_{Lab}$, we have that for all labels, either

- there exists a jump corresponding to the label that is continuously enabled in the interval $(0,\tau)$, which has the effect that the valuation of the label is decreased by τ, or
- at no point within these τ time units, any jump associated with the label is enabled. Then, the valuation remains unchanged.

Let $e = (id,l,\mu,l') \in Edge$ with $r = Proc(e)$. Considering the first case in more detail, the condition expresses that for all valuations ν' reached by letting τ time units pass, the edge e is still enabled in the state $\vartheta' = (l,\nu',\nu_{Lab})$. Dually in the second case, the edge e must be disabled in all intermediate states.

Definition 2.12. Operational Semantics of SHAs

Let $\mathcal{A}$ be an SHA. The operational semantics of $\mathcal{A}$ is given by the following three rules:

$$\frac{f \in Act(l) \quad \tau \in \mathbb{R}_{>0} \quad \nu' = time^f_\vartheta(\tau) \quad \nu'_{Lab} = clocks^f_\vartheta(\tau)}{(l,\nu,\nu_{Lab}) \xrightarrow{\tau,f} (l,\nu',\nu'_{Lab})} \text{ time}$$

$$\frac{e = (id,l,\mu,l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e),(l',\nu')))}{(l,\nu,\nu_{Lab}) \xRightarrow{e} (l',\nu',\nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

$$\frac{\nu \approx_{Con} \nu'}{(l,\nu,\nu_{Lab}) \xRightarrow{\text{env}} (l,\nu',\nu_{Lab})} \text{ environment}$$

One *execution step*, denoted by $\Rightarrow$, is taken by using one of the above rules, i.e., we define

$$\Rightarrow \quad = \quad \Big(\bigcup_{\tau\in\mathbb{R}_{>0}, f\in\mathcal{F}} \xrightarrow{\tau,f}\Big) \cup \Big(\bigcup_{e\in Edge} \xRightarrow{e}\Big) \cup \xrightarrow{\text{env}}$$

State ϑ_i	Location	Variables ν_i	Labels $\nu_{Lab,i}$
ϑ_0	l_0	$x \mapsto 0$	$a \mapsto 1, b \mapsto 4$
ϑ_1	l_0	$x \mapsto 6$	$a \mapsto 1, b \mapsto 4$
ϑ_2	l_0	$x \mapsto 8$	$a \mapsto 0, b \mapsto 4$
ϑ_3	l_1	$x \mapsto 8$	$a \mapsto 2, b \mapsto 4$
ϑ_4	l_1	$x \mapsto 6$	$a \mapsto 2, b \mapsto 4$
ϑ_5	l_1	$x \mapsto 2$	$a \mapsto 2, b \mapsto 0$
ϑ_6	l_0	$x \mapsto 0$	$a \mapsto 2, b \mapsto 4$

Table 2.1.: A path of $\mathcal{A}_1$ from Example 2.1. The semantic steps taken to get from ϑ_i to ϑ_{i+1} are discussed in Example 2.2.

A *path* of $\mathcal{A}$ is an infinite sequence $\vartheta_0, \vartheta_1, \vartheta_2, \ldots$ of states of $\mathcal{A}$ such that

$$\vartheta_0 \Rightarrow \vartheta_1 \Rightarrow \vartheta_2 \Rightarrow \ldots$$

and ϑ_0 is an initial state. Under abuse of notation we also write $\vartheta \Rightarrow \vartheta'$ if ϑ' is reachable from ϑ taking several execution steps.

Hence, there are three possible steps we can take: First, time steps describe what happens when τ time units pass. The location remains unchanged, and the values of the variables and the clocks evolve as described by the predicates above. Second, we can take a discrete step following a jump e, given that e is fireable. In this case, the location is modified, and the variables are reset as specified by the jump. Additionally, we re-sample the enabling duration and set the corresponding clock variable accordingly. Third, we can take environmental steps that manipulate the non-controlled variables.

EXAMPLE 2.2. **Semantics of SHAs**
Consider again the SHA $\mathcal{A}_1$ discussed in Example 2.1. A finite prefix of a path of $\mathcal{A}_1$ is given in Table 2.1, where we have that

$$\vartheta_0 \xrightarrow{\tau = 3} \vartheta_1 \xrightarrow{\tau = 1} \vartheta_2 \xrightarrow{e_a} \vartheta_3 \xrightarrow{\tau = 2} \vartheta_4 \xrightarrow{\tau = 4} \vartheta_5 \xrightarrow{e_b} \vartheta_6.$$

We do not explicitly denote the activation function, as it is unique for x in every location. The initial state θ_0 is given by the initial location l_0 and the valuation of x to zero. Additionally, we sample values for the valuation of the labels a and b. In this example, assume that we sample one for a. For b, four is the only possible value because we are sampling from the Dirac distribution δ_4.

From ϑ_0, we take a time step of length three, so $\vartheta_0 \xRightarrow{\tau = 3} \vartheta_1$. The activity function is given by $f_0(\tau) = v_0(x) + 2 \cdot \tau$. Therefore, x is increased by $2 \cdot 3 = 6$. The valuation of the labels does not change, as no corresponding jump is enabled. In ϑ_1, however, the condition for e_a is fulfilled: We have $x \geq 6$. Therefore, when another time unit passes, we reach ϑ_2 by $\vartheta_1 \xRightarrow{\tau = 1} \vartheta_2$, where x is again increased by two and a is decreased by one.

Now, we have to take a jump, as the next time step would lead to x being greater than eight, which violates the invariant of l_0. Fortunately, the jump e_a is fireable. Thus, we take e_a and have $\vartheta_2 \xRightarrow{e_a} \vartheta_3$.

In ϑ_3, the location changed to l_1, and we sampled a new value for a, in this case two. Then, we take a time step of length two, $\vartheta_3 \xRightarrow{\tau = 2} \vartheta_4$, and reach a state where x is decreased by two and the labels remain the same as the guard of the only available jumps is violated. In the next time step $\vartheta_4 \xRightarrow{\tau = 4} \vartheta_5$, b is decreased as the guard of e_b is fulfilled.

Finally, we take the jump e_b and are in location l_0 again, so $\vartheta_5 \xRightarrow{e_b} \vartheta_6$. As specified by the effect of the jump, x is reset to zero. Also, we again sample from δ_4 and set b to four.

Conflict Resolution. To get a full probabilistic model, one has to resolve conflicts, which are arising, for example, when several jumps become enabled at the same time. Therefore, we have to define a scheduler that decides which jump to take. While all approaches are conceivable, in this thesis we choose to resolve conflicts probabilistically using the weight assigned to jumps. More precisely, let e be a fireable jump in state σ. Then, the probability of taking e is given by

$$\frac{Wgt(e)}{\sum_{e' \in Edge,\ fireable_\sigma(e')} Wgt(e')}.$$

Here, it also becomes evident that we need to define the weights to be positive to avoid division by zero.

Unrealistic Behavior. It might happen that an SHA behaves unexpectedly or unrealistically. For example, it can be possible to take infinitely many steps within finite time. Also, time can stop when no jump is fireable and the invariant would be violated by further time steps. This is illustrated in the following example.

EXAMPLE 2.3. **Unrealistic Behavior of SHAs**

We continue Example 2.2. From $\vartheta_6 = (l_0, (x \mapsto 0), (a \mapsto 2, b \mapsto 4))$ we can only take a time step. Let ϑ_7 be the state reached after letting three time units pass, i.e.,

$$\vartheta_6 \xrightarrow{\tau = 3} \vartheta_7 = (l_0, (x \mapsto 6), (a \mapsto 2, b \mapsto 4)).$$

Now, e_a is enabled, so we take a time step and the value of a is decreased accordingly:

$$\vartheta_7 \xrightarrow{\tau = 1} \vartheta_8 = (l_0, (x \mapsto 8), (a \mapsto 1, b \mapsto 4)).$$

This leaves us in a state where we cannot take any more discrete or time steps: Time steps are not possible as this would violate the invariant $x \leq 8$ of l_0. We can also not take the jump e_a, as the corresponding clock value is not zero yet. Unless the value sampled for a is between zero and one, we will always end up in such a state.

Example 2.3 shows how quickly unrealistic behavior can occur. We can distinguish between three types of unwanted behavior. The first are *Zeno paths*, which are infinite paths that only take finite time, but infinitely many discrete steps are taken within the path. Clearly, this cannot happen in the real world.

Next, a state can have a *timelock*. Intuitively, this means that from this state, all paths of infinite length only take finite time. A *deadlock* is a special case of a timelock where it depends exclusively on the environment if a path can be taken from a state. This was the case in Example 2.3.

Avoiding such behavior is, in general, the task of the modeler. We will define the transformation in such a way that it fulfills this task. Therefore, we do not give more details here and instead refer to [Ger22, Section 2.2.1, 3.2.2] for both formalisms and examples of these challenges.

2.3. Composition

One of our goals for the transformation is to define it in a composable way. Therefore, we now define how two SHAs can be composed. We start by giving a criterion for whether two SHAs can be composed.

DEFINITION 2.13. **Composability of SHA**

Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two SHAs. We say that $\mathcal{A}_1$ and $\mathcal{A}_2$ are *composable* if

- $Var_1 = Var_2$,
- $Con_1 \cap Con_2 = \emptyset$,

- for all $(id_1, l_1, \mu_1, l_1') \in Edge_1$ and $(id_2, l_2, \mu_2, l_2') \in Edge_2$ we have $id_1 \neq id_2$, and
- $Lab_1 \cap Lab_2 = \emptyset$.

In contrast to Gerlach, we require the set of variables to coincide [Ger22]. If this is not enforced, one can define an extension of SHAs and define the composition with respect to these extensions as done by Gerlach. Since we will only consider the composition of SHAs defined over the same set of variables, we omit this and instead require the variable sets to coincide.

Additionally, each variable can only be controlled by one SHA, and the identities of the jumps must be unique. The set of labels must also be disjoint. Then, we can define the composition of two SHAs as follows:

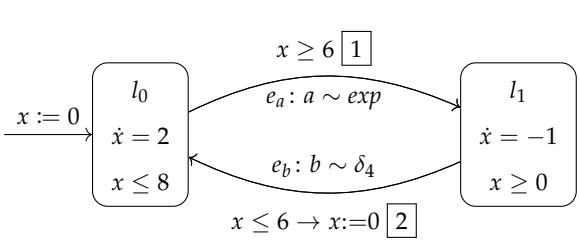
DEFINITION 2.14. **Syntactic Parallel Composition of SHAs**

Let $\mathcal{A}_1, \mathcal{A}_2$ be two composable SHAs. Their *parallel composition* is the SHA

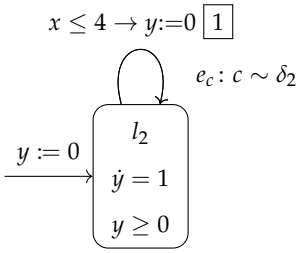
$$\mathcal{A}_1 \parallel \mathcal{A}_2 = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt)$$

with

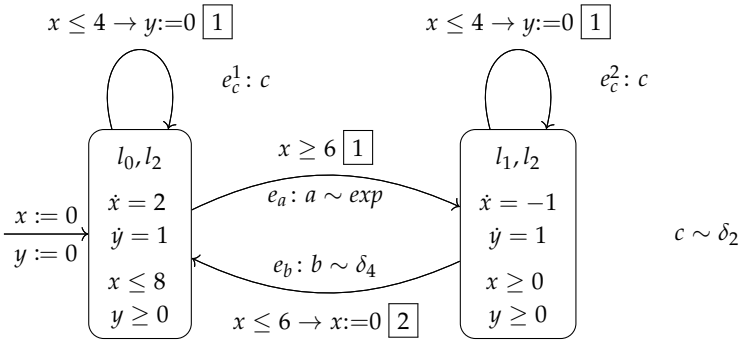
- $Loc = Loc_1 \times Loc_2$.
- $Var = Var_1 = Var_2$ with $Con = Con_1 \cup Con_2$ and $NCon = Var \backslash Con$.
- $Inv(l_1, l_2) = Inv_1(l_1) \cap Inv_2(l_2)$ for all $(l_1, l_2) \in Loc$.
- $Init(l_1, l_2) = Init_1(l_1) \cap Init_2(l_2)$ for all $(l_1, l_2) \in Loc$.
- $(id, (l_1, l_2), \mu, (l_1', l_2')) \in Edge$ iff
  - $l_2 = l_2'$ and there exists $(id, l_1, \mu, l_1') \in Edge_1$, or
  - $l_1 = l_1'$ and there exists $(id, l_2, \mu, l_2') \in Edge_2$.
- $Act(l_1, l_2) = Act_1(l_1) \cap Act_2(l_2)$ for all $(l_1, l_2) \in Loc$.
- $Lab = Lab_1 \cup Lab_2$
- $Proc(id, (l_1, l_2), \mu, (l_1', l_2')) = \begin{cases} Proc_1(e_1) & \text{if } e_1 = (id, l_1, \mu, l_1') \in Edge_1 \wedge l_2 = l_2' \\ Proc_2(e_2) & \text{if } e_2 = (id, l_2, \mu, l_2') \in Edge_2 \wedge l_1 = l_1'. \end{cases}$
- $Dur(r, ((l_1, l_2), \nu)) = \begin{cases} Dur_1(r, (l_1, \nu)) & \text{if } r \in Lab_1 \\ Dur_2(r, (l_2, \nu)) & \text{if } r \in Lab_2. \end{cases}$
- $Wgt(id, (l_1, l_2), \mu, (l_1', l_2')) = \begin{cases} Wgt_1(e_1) & \text{if } e_1 = (id, l_1, \mu, l_1') \in Edge_1 \wedge l_2 = l_2' \\ Wgt_2(e_2) & \text{if } e_2 = (id, l_2, \mu, l_2') \in Edge_2 \wedge l_1 = l_1'. \end{cases}$



(a) Depiction of the SHA $\mathcal{A}_1$ from Example 2.1.



(b) Depiction of the SHA $\mathcal{A}_2$.



(c) Depiction of the composed SHA $\mathcal{A}_1 \parallel \mathcal{A}_2$.

Figure 2.2.: Composition of SHAs $\mathcal{A}_1$ and $\mathcal{A}_2$.

The SHA $\mathcal{A}_1 \parallel \mathcal{A}_2$ intuitively executes both $\mathcal{A}_1$ and $\mathcal{A}_2$ in parallel. Invariants and initial conditions are combined, which formally corresponds to an intersection of the respective sets. A jump in the composition always originates from either $\mathcal{A}_1$ or $\mathcal{A}_2$ and has the same guard and effect as it did in the original SHA. The location of the other SHA remains unchanged. The activities are common activities from both sub-SHAs. For the probabilistic components, the composed SHA always chooses the appropriate part from the sub-SHA from which the jump or label originates.

EXAMPLE 2.4. **Composition of SHAs**

Consider the composition of SHAs $\mathcal{A}_1$ and $\mathcal{A}_2$ depicted in Figure 2.2. We already discussed $\mathcal{A}_1$ as depicted in Figure 2.2a in earlier examples in detail, so we do not repeat this here and instead refer to Example 2.1.

$\mathcal{A}_2$ as depicted in Figure 2.2b is made of only one location l_0 and controls a variable y. While in location l_0, y is increased by one per time unit. There is one jump e_c that can be taken when x is less than or equal to four. The assigned distribution is a Dirac distribution, so we take this jump after it was enabled for two time units. When e_c is taken, y is reset to zero.

The formal definition of $\mathcal{A}_2$ is given by

$$\mathcal{A}_2 = (Loc_2, Var_2, Inv_2, Init_2, Edge_2, Act_2, Lab_2, Proc_2, Dur_2, Wgt_2),$$

where:

- $Loc_2 = \{l_2\}$
- $Var_2 = \{x, y\}$ with $Con_2 = \{y\}$ and $NCon_2 = \{x\}$
- $Inv_2(l_2) = \{\nu \in \mathcal{V} \mid \nu(y) \geq 0\}$
- $Init_2(l_2) = \{\nu \in \mathcal{V} \mid \nu(y) = 0\}$
- $Edge_2 = \{e_c\}$ with $e_c = (l_2, \mu_c, l_2)$
  and $\mu_c = \{(\nu, \nu') \mid \nu(x) \leq 4 \wedge \nu'(y) = 0 \wedge \nu'(x) = \nu(x)\}$
- $Act_2(l_2)$ is the set of activities that are solutions of $\dot{y} = 1$
- $Lab_2 = \{c\}$
- $Proc_2(e_c) = c$
- $Dur_2(c, (l_2, \nu)) = \delta_2$ for all $\nu \in \mathcal{V}$
- $Wgt_2(e_c) = 1$

First, note that $\mathcal{A}_1$ and $\mathcal{A}_2$ are indeed composable as defined in Definition 2.13. A depiction of their composition is given in Figure 2.2. Formally,

$$\mathcal{A}_1 \parallel \mathcal{A}_2 = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt)$$

is the SHA with

- $Loc = Loc_1 \times Loc_2 = \{(l_0, l_2), (l_1, l_2)\}$
- $Var = \{x, y\}$ with $Con = Con_1 \cup Con_2 = \{x, y\}$ and $NCon = \emptyset$
- $Inv(l_0, l_2) = Inv_1(l_0) \cap Inv_2(l_1) = \{\nu \in \mathcal{V} \mid \nu(x) \leq 8 \wedge \nu(y) \geq 0\}$
  $Inv(l_1, l_2) = Inv_1(l_1) \cap Inv_2(l_1) = \{\nu \in \mathcal{V} \mid \nu(x) \geq 0 \wedge \nu(y) \geq 0\}$
- $Init(l_0, l_2) = Init_1(l_0) \cap Init_2(l_2) = \{\nu \in \mathcal{V} \mid \nu(x) = \nu(y) = 0\}$
  $Init(l_1, l_2) = \emptyset$
- $Edge = \{e_a, e_b, e_c^1, e_c^2\}$ with
  - $e_a = ((l_0, l_2), \mu_a, (l_1, l_2))$
    and $\mu_a = \{(\nu, \nu') \mid \nu(x) \geq 6 \wedge \nu'(y) = \nu(y)\}$
  - $e_b = ((l_1, l_2), \mu_b, (l_0, l_2))$
    and $\mu_b = \{(\nu, \nu') \mid \nu(x) \leq 6 \wedge \nu'(x) = 0 \wedge \nu'(y) = \nu(y)\}$

  - $e_c^1 = ((l_0, l_2), \mu_c, (l_0, l_2))$
    and $\mu_c = \{(\nu, \nu') \mid \nu(x) \leq 4 \wedge \nu'(y) = 0 \wedge \nu'(x) = \nu(x)\}$
  - $e_c^2 = ((l_1, l_2), \mu_c, (l_1, l_2))$

- $Act(l_0, l_2) = Act_1(l_1) \cap Act_2(l_2)$, which is the set of activities that are solutions of $\dot{x} = 2$ and $\dot{y} = 1$.

  $Act_1(l_2, l_3)$ is the set of activities that are solutions of $\dot{x} = -1$ and $\dot{y} = 1$.
- $Lab = Lab_1 \cup Lab_2 = \{a, b, c\}$
- $Proc(e_a) = a$, $Proc(e_b) = b$, and $Proc(e_c^1) = Proc(e_c^2) = c$
- $Dur(a, (l, \nu)) = e^{-t}$, $Dur(b, (l, \nu)) = \delta_4$, and $Dur(c, (l, \nu)) = \delta_2$ for all $\nu \in \mathcal{V}$, $l \in Loc$
- $Wgt(e_a) = 1$, $Wgt(e_b) = 2$, $Wgt(e_c^1) = Wgt(e_c^2) = 1$

This example also illustrates how weights are used to resolve conflicts. Assume we are in a state (loc, ν, ν_{Lab}) with location $loc = (l_1, l_2)$, $\nu(x) = 4$ and $\nu_{Lab}(b) = \nu_{Lab}(c) = 0$. Then, both jumps e_b and e_c^2 are fireable and thus in conflict. Since $Wgt(e_b) = 2$ and $Wgt(e_c^2) = 1$, jump e_b will be taken with probability $\frac{2}{3}$.

2.4. Assumptions and Restrictions

The definitions in this chapter largely agree with those given by Gerlach in [Ger22]. There are some minor differences that will be summarized in what follows.

First, we define the SHA without providing a formal definition of hybrid automata beforehand. As a consequence, we cannot build on concepts also existing for hybrid automata. For example, Definitions 2.8 and 2.10 essentially describe the semantics of hybrid automata. Also, we renamed the predicate *invEn* from [Ger22, Definition 3.2.2] to *clocks* (Definition 2.11), but they express the same concept. In conclusion, while the formalism varies slightly, the semantics agree in terms of content and describe the same set of paths.

For the composition, we add the requirement that the set of variables of composable SHAs has to coincide (see Definition 2.13). As discussed in Section 2.3, the reason for this is that we only compose SHAs defined over the same set of variables, so this is always given in the transformation. This simply has the advantage that we can save some notation. Other than that, the definition of the composition is equivalent.

For the calculation of path probabilities, Gerlach assumes that there are no invariants, in order to avoid deadlocks. In the transformation presented in this thesis, we will

construct SHAs in a way that prevents deadlocks by always providing a jump that can be taken in case an invariant is violated. In this way, we cannot end up in a deadlock and can allow invariants.

3. Hybrid Petri Nets with General Firings

Petri nets were proposed presumably in the mid-twentieth century by Carl Adam Petri for modeling chemical processes. Like hybrid automata, they can be viewed as an extension of discrete automata. A Petri net is also made of locations, now called places, connected via transitions. A place can hold a number of tokens, which are moved between places when a transition fires. A state is then characterized by the number of tokens contained in each place, essentially a set of discrete variables. This allows for a more precise description of processes compared to standard automata. Moreover, Petri nets have an intuitive graphical representation.

To model hybrid systems, Petri nets were extended to include both discrete and continuous variables [AA97; DA01; DA10]. In a hybrid Petri net, places as well as transitions are split into discrete and continuous parts.

Finally, stochastic components were included to account for uncertainties. Early approaches include generalized stochastic Petri nets [ACB84], fluid stochastic Petri nets [TK93], or stochastic activity networks [SM00]. In this thesis, we focus on a specific subclass of stochastic hybrid Petri nets introduced by Gribaudo and Remke [GR10; GR16]. In a *hybrid Petri net with general firings* (HPnG), stochasticity is introduced in the form of transitions that fire according to some probability distribution.

As in the previous chapter, we start by defining the syntax of HPnGs in Section 3.1 and their semantics in Section 3.2. In particular, we discuss the resolution of conflicts in Section 3.2.1 and Section 3.2.2 for discrete and continuous transitions, respectively, proposing a novel rate adaptation algorithm in the latter.

3.1. Syntax of HPnGs

HPnGs essentially consist of places and transitions connected by arcs. We distinguish for all three between discrete and continuous components. Additionally, the syntax includes specifications for the initial states, in the form of initial markings, and a set of parameters for further refinement.

Definition 3.1. Hybrid Petri Nets with General Firings

A *hybrid Petri net with general firings* (HPnG) is a tuple $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$, where:

- $\mathcal{P} = \mathcal{P}^{disc} \dot\cup \mathcal{P}^{cont}$ is a finite set of places, partitioned into disjoint sets of discrete and continuous places.
- $\mathcal{T} = \mathcal{T}^{det} \dot\cup \mathcal{T}^{gen} \dot\cup \mathcal{T}^{cont}$ is a finite set of transitions, partitioned into disjoint sets of deterministic, general, and continuous transitions. We write $\mathcal{T}^{disc} = \mathcal{T} \backslash \mathcal{T}^{cont}$ for the set of discrete transitions.
- $\mathcal{A} = \mathcal{A}^{disc} \dot\cup \mathcal{A}^{cont} \dot\cup \mathcal{A}^{test} \dot\cup \mathcal{A}^{inh}$ is a finite set of arcs, partitioned into disjoint sets of discrete, continuous, test, and inhibitor arcs. Every arc connects one place and one transition of specified types:
  - $\mathcal{A}^{disc} \subseteq \left(\mathcal{P}^{disc} \times \mathcal{T}^{disc}\right) \cup \left(\mathcal{T}^{disc} \times \mathcal{P}^{disc}\right)$,
  - $\mathcal{A}^{cont} \subseteq \left(\mathcal{P}^{cont} \times \mathcal{T}^{cont}\right) \cup \left(\mathcal{T}^{cont} \times \mathcal{P}^{cont}\right)$,
  - $\mathcal{A}^{test} \subseteq \mathcal{P} \times \mathcal{T}$, and
  - $\mathcal{A}^{inh} \subseteq \mathcal{P} \times \mathcal{T}$.
- $M_0 = (m_0, x_0)$ is an initial marking with $m_0 \colon \mathcal{P}^{disc} \to \mathbb{N}_0$ and $x_0 \colon \mathcal{P}^{cont} \to \mathbb{R}_{\geq 0}$
- $\Phi = \left(\Phi^{\mathcal{P}}_{ub}, \Phi^{\mathcal{T}}_{ft}, \Phi^{\mathcal{T}}_{gt}, \Phi^{\mathcal{T}}_{fr}, \Phi^{\mathcal{A}}_{w}, \Phi^{\mathcal{T}}_{p}\right)$ is a tuple of parameter functions with
  - $\Phi^{\mathcal{P}}_{ub} \colon \mathcal{P}^{cont} \to (\mathbb{R}_{>0} \cup \infty)$, defining an upper bound on the marking of every continuous place,
  - $\Phi^{\mathcal{T}}_{ft} \colon \mathcal{T}^{det} \to \mathbb{R}_{>0}$, defining a firing time of every deterministic transition,
  - $\Phi^{\mathcal{T}}_{gt} \colon \mathcal{T}^{gen} \to \mathit{Distr}$, assigning a probability distribution to every general transition,
  - $\Phi^{\mathcal{T}}_{fr} \colon \mathcal{T}^{cont} \to \mathbb{R}_{>0}$, defining a nominal firing rate for every continuous transition,
  - $\Phi^{\mathcal{A}}_{w} \colon \mathcal{A} \backslash \mathcal{A}^{cont} \to \mathbb{R}_{>0}$, defining a weight for every non-continuous arc, and
  - $\Phi^{\mathcal{T}}_{p} \colon \mathcal{T}^{disc} \to \mathbb{N}_{>0}$, defining a priority for every discrete transition.

A visualization of the basic components is given in Figure 3.1. We now give an intuitive explanation for each component of the HPnG.

Places. There are two different types of places in a HPnG: Discrete places holding a discrete number of tokens, and continuous places holding some continuous amount

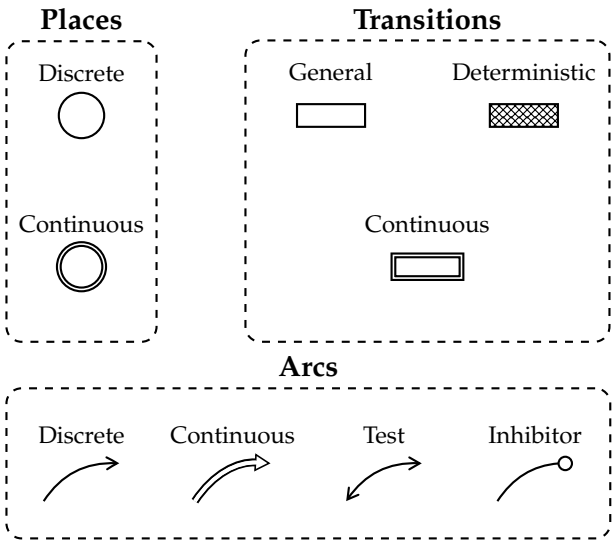


Figure 3.1.: Depiction of the primitives of a HPnG based on [GR16]. Continuous components are generally distinguished from discrete components by the use of double lines. Additionally, deterministic transitions are marked by a filled rectangle, while general transitions are empty. Test arcs have an arrow in both directions and inhibitor arcs have a small circle at the end connected to the transition.

of fluid. For example, we can use discrete places to model switches and continuous places to model water tanks.

Discrete places can hold any non-negative discrete number of tokens. For continuous places, a restriction on the amount of fluid a place p can hold is given by an implicit lower bound of zero and a not necessarily finite upper bound $\Phi_{ub}^{\mathcal{P}}(p)$. The number of tokens respectively the amount of fluid of a place is referred to as the discrete or continuous *marking*.

The model could be extended by also defining an upper bound for discrete places. This requires some straightforward adaptions in the semantics and consequently also in the definition of the transformation. For the sake of brevity, we do not assume upper bounds on the discrete places.

Transitions. A transition has *concession* when the connected places contain a specified amount of tokens or fluid. More concretely, the weight of the arc between a transition and a place, given by $\Phi_w^{\mathcal{A}}$, controls how many tokens or how much fluid is required, as will be explained in detail at a later point in Section 3.2. When a transition has concession for a specified time, we say that it is *fireable* and can thus *fire*, which means that it moves tokens or fluid from its input place to its output place. The exact number of tokens moved depends again on the weight of the arc that connects the transition to

the place.

Discrete transitions are also split into general and deterministic transitions. Deterministic transitions fire after having concession for the number of time units specified by the parameter function $\Phi_{ft}^{\mathcal{T}}$. General transitions fire after a number of time units following a continuous probability distribution specified by $\Phi_{gt}^{\mathcal{T}}$. This concept of delaying the firing for a certain time is very similar to the enabling duration of jumps in SHAs (see Section 2.1). It is not required that the transition has concession continuously, i.e., if the transition does not have concession anymore, its clock is not reset. We could additionally distinguish transitions that fire immediately [Gha17; GR16]. However, for simplicity, we regard those as a special case of a deterministic transition t where $\Phi_{ft}^{\mathcal{T}}(t)$ is zero.

Continuous transitions fire as soon as they have concession, comparable to discrete transitions with a firing time of zero. However, they do not fire once, but continuously. The firing rate $\Phi_{fr}^{\mathcal{T}}$ specifies the *nominal firing rate* of each transition, i.e., how much fluid is moved per time unit. Since continuous places might not be able to hold an infinite amount of fluid, the firing rate potentially has to be adapted in order not to exceed the place's bounds. Dually, the rate has to be adapted if a transition wants to move more fluid from a place than it currently holds. The *rate adaption* algorithm used to determine new firing rates for the affected transitions is described in more detail in Section 3.2.2. We therefore distinguish the nominal firing rate from the *actual firing rate* that depends on the current state.

Arcs. Arcs connect transitions to places. We denote an arc as $\langle t, p \rangle$ or $\langle p, t \rangle$ for a transition $t \in \mathcal{T}$ and a place $p \in \mathcal{P}$. There are different types of arcs, namely discrete, fluid, inhibitor or test arcs, that differ in the types of transitions and places they can connect and in their effects on the system. Between one place and one transition, there can be at most one arc.

Discrete arcs connect discrete places to discrete transitions. Their assigned weight $\Phi_{w}^{\mathcal{A}}$ corresponds to the number of tokens that the place has to hold for the transition to have concession and that are moved when the transition fires. Continuous arcs connect continuous places to continuous transitions. They could also be weighted to allow for a more versatile calculation of the actual firing rate. For simplicity, we refrain from doing so and only assign weights to non-continuous arcs.

Inhibitor and test arcs can connect all combinations of transitions and places. In particular, they are used to connect the discrete and continuous fragments of a HPnG. In contrast to the other arcs, they are semantically undirected. The weight of a test arc specifies the number of tokens or the amount of fluid that a place has to hold for the connected transition to have concession. On the other hand, inhibitor arcs block the transition from firing when the tokens or fluid surpass the assigned weight of the arc.

Notice that test and inhibitor arcs both do not affect the marking and only influence the enabling status of the transition.

Parameter Functions. Parameters can be set to provide a more detailed description of the system being modeled. In the previous descriptions we have already mentioned the purpose of most of the parameters. This is now summarized and extended in the following list.

- $\Phi^{\mathcal{P}}_{ub}$ specifies the *upper bound* on the amount of fluid a continuous place can hold.
- $\Phi^{\mathcal{T}}_{ft}$ specifies the *firing time* of every deterministic transition, i.e., the time it needs to have concession before firing.
- $\Phi^{\mathcal{T}}_{gt}$ assigns to each general transition a cumulative continuous *probability distribution* that determines when the transition fires.
- $\Phi^{\mathcal{T}}_{fr}$ specifies the *nominal firing rate* of the continuous transitions.
- $\Phi^{\mathcal{A}}_{w}$ assigns a *weight* to each non-continuous arc, which corresponds to the number of tokens or amount of fluid required by the arc. If connected to a discrete place, the weight of the arc must be a natural number. For better readability, we write $\Phi^{\mathcal{A}}_{w}\langle p, t\rangle$ instead of $\Phi^{\mathcal{A}}_{w}(\langle p, t\rangle)$.
- $\Phi^{\mathcal{T}}_{p}$ assigns a *priority* to each discrete transition. As the weights assigned to jumps of SHAs, these will be used to resolve conflicts when two transitions want to fire at the same time.

In the literature, the parameter functions vary depending on the application. For example, Gribaudo et al. assign weights and priorities to discrete transitions; and shares and priorities to continuous transitions for the purpose of conflict resolution [GR16]. We simplify this by assigning priorities only to discrete transitions, which is adequate for the conflict resolution method proposed in Section 3.2. Another minor difference, already mentioned above, is that we do assign weights to continuous arcs. This is done in order to better regulate the actual firing rate of continuous transitions. However, our definitions can be extended to include all of these parameters in a straightforward manner.

In general, there is a wide scope for adapting the parameters to a particular application. For example, one could assume an upper bound not only for continuous places but also for discrete places. Since this thesis is the first to define a direct transformation between SHAs and HPnGs, we decided to keep the parameters as simple as possible to focus on the transformation and avoid unnecessarily long proofs. Although we make some simplifying assumptions, the given framework remains very flexible and can be easily adapted.

As already indicated above, a HPnG can be separated into discrete and continuous parts. In particular, we sometimes speak of the *discrete fragment* of a HPnG, which refers to a set of discrete places that are connected via transitions.

Definition 3.2. Discrete Fragments

Let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG. A set of discrete places $\mathcal{D} \subseteq \mathcal{P}^{disc}$ is called a *discrete fragment* of $\mathcal{H}$ if

- all pairs of places in $\mathcal{D}$ are connected, i.e., for all places $p, p' \in \mathcal{D}$ exist places $p_1, p_2, \ldots, p_n \in \mathcal{D}$ and transitions $t_1, \ldots, t_{n-1} \in \mathcal{T}$ such that $p = p_1$, $p' = p_n$ and

$$\langle p_i, t_i \rangle, \langle t_i, p_{i+1} \rangle, \langle t_{n-1}, p_n \rangle \in \mathcal{A} \text{ for all } i \in \{1, \ldots, n-1\}, \text{ and}$$

- $\mathcal{D}$ is maximal, i.e., there does not exist a place $p \in \mathcal{P} \backslash \mathcal{D}, p' \in \mathcal{D}$ and a transition $t \in \mathcal{T}$ such that $\langle p, t \rangle, \langle t, p' \rangle \in \mathcal{A}$.

We denote by $\mathcal{T}^{\mathcal{D}}$ the transitions that are included in the discrete fragment, i.e.,

$$\mathcal{T}^{\mathcal{D}} = \left\{ t \in \mathcal{T}^{disc} \;\middle|\; \exists p \in \mathcal{D}.\, (\langle p, t \rangle \in \mathcal{A} \vee \langle t, p \rangle \in \mathcal{A}) \right\},$$

and by $\mathcal{A}^{\mathcal{D}}$ the arcs that are included in the discrete fragment, i.e.,

$$\mathcal{A}^{\mathcal{D}} = \left\{ \langle p, t \rangle, \langle t, p \rangle \in \mathcal{A} \;\middle|\; t \in \mathcal{T}^{\mathcal{D}} \right\}.$$

Notice that all discrete components of a HPnG uniquely belong to one discrete fragment.

Example 3.1. Syntax of HPnGs

Consider the HPnG $\mathcal{H}$ depicted in Figure 3.2 modeling an irrigation system. We assume that the units are given by liters and hours, for example, firing rates are given in liters per hour. The general transition t_{on} moves tokens to the discrete place $p_{rainmaker}$, and another general transition t_{off} removes the tokens again. When there are more than two tokens in $p_{rainmaker}$, the continuous transition t_{rain} has concession, modeling the rain and filling a rain barrel with a nominal firing rate of ten liters per hour. The barrel is modeled by the continuous place p_{barr} that holds at most a hundred liters. Unfortunately, the barrel leaks, modeled by the transition t_{leak} removing one liter per hour from the barrel. The barrel also provides water for an irrigation system t_{irr}, which removes twenty liters of water per hour from the barrel to some plants modeled by p_{plant}. The irrigation system is controlled by the place $p_{control}$. If there are tokens in $p_{control}$, the system is blocked, meaning in particular that t_{irr} does not fire. The control depends on a choice mechanism: The general

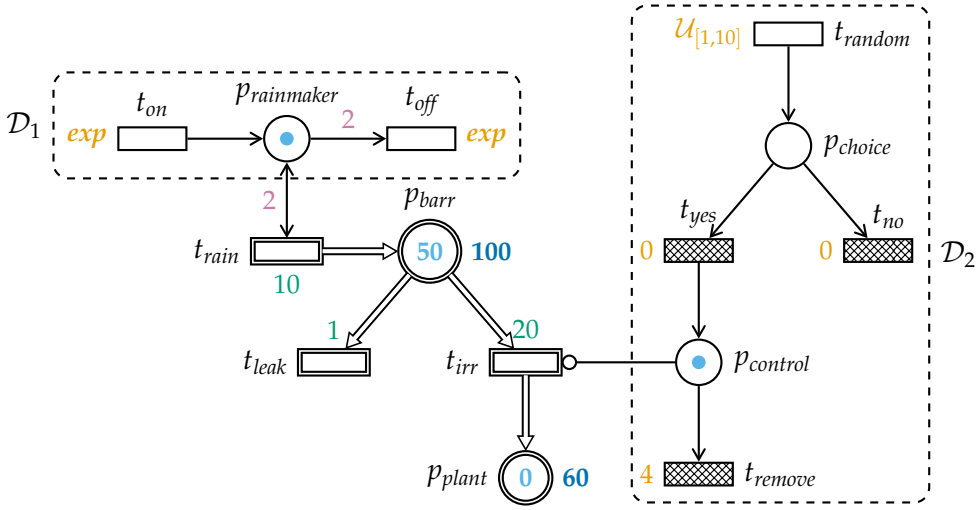


Figure 3.2.: The HPnG $\mathcal{H}$ modeling an irrigation system as discussed in Example 3.1. The components are depicted as described in Figure 3.1. We denote initial markings in the places. Parameters are denoted next to the relevant components: Upper boundaries for continuous places, firing times for deterministic transitions, probability distributions for general transitions, and firing rates for continuous transitions. The weights of the arcs are one unless denoted otherwise. Since the priorities for all transitions are set to one in this example, we omit them. The discrete fragments $\mathcal{D}_1$ and $\mathcal{D}_2$ are grouped by the dashed lines.

transition t_{random} moves tokens to a place p_{choice}, where two transitions t_{yes} and t_{no} compete and consequently decide whether the tokens are moved to $p_{control}$ or not.

Formally, $\mathcal{H}$ is given by $(\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$, where:

- $\mathcal{P} = \mathcal{P}^{disc} \dot\cup \mathcal{P}^{cont}$ with
  - $\mathcal{P}^{disc} = \{p_{rainmaker}, p_{control}, p_{choice}\}$ and
  - $\mathcal{P}^{cont} = \{p_{barr}, p_{plant}\}$.
- $\mathcal{T} = \mathcal{T}^{det} \dot\cup \mathcal{T}^{gen} \dot\cup \mathcal{T}^{cont}$ with
  - $\mathcal{T}^{det} = \{t_{remove}, t_{yes}, t_{no}\}$,
  - $\mathcal{T}^{gen} = \{t_{on}, t_{off}, t_{random}\}$, and
  - $\mathcal{T}^{cont} = \{t_{rain}, t_{leak}, t_{irr}\}$.
- $\mathcal{A} = \mathcal{A}^{disc} \dot\cup \mathcal{A}^{cont} \dot\cup \mathcal{A}^{test} \dot\cup \mathcal{A}^{inh}$ with
  - $\mathcal{A}^{disc} = \{\langle t_{on}, p_{rainmaker}\rangle, \langle p_{rainmaker}, t_{off}\rangle, \langle p_{control}, t_{remove}\rangle, \langle t_{yes}, p_{control}\rangle, \langle p_{choice}, t_{yes}\rangle, \langle p_{choice}, t_{no}\rangle, \langle t_{random}, p_{choice}\rangle\}$,

  - $\mathcal{A}^{cont} = \{\langle t_{rain}, p_{barr}\rangle, \langle p_{barr}, t_{leak}\rangle, \langle p_{barr}, t_{irr}\rangle, \langle t_{irr}, p_{plant}\rangle\}$,
  - $\mathcal{A}^{test} = \{\langle p_{rainmaker}, t_{rain}\rangle\}$, and
  - $\mathcal{A}^{inh} = \{\langle p_{control}, t_{irr}\rangle\}$.
- $M_0 = (m_0, x_0)$ with
  - $m_0(p_{rainmaker}) = 1$,
  - $m_0(p_{control}) = 1$,
  - $m_0(p_{choice}) = 0$, and
  - $x_0(p_{barr}) = 50$,
  - $x_0(p_{plant}) = 0$.
- $\Phi = (\Phi^{\mathcal{P}}_{ub}, \Phi^{\mathcal{T}}_{ft}, \Phi^{\mathcal{T}}_{gt}, \Phi^{\mathcal{T}}_{fr}, \Phi^{\mathcal{A}}_{w}, \Phi^{\mathcal{T}}_{p})$ with
  - $\Phi^{\mathcal{P}}_{ub}(p_{barr}) = 100, \Phi^{\mathcal{P}}_{ub}(p_{plant}) = 60$,
  - $\Phi^{\mathcal{T}}_{ft}(t_{remove}) = 4, \Phi^{\mathcal{T}}_{ft}(t_{yes}) = \Phi^{\mathcal{T}}_{ft}(t_{no}) = 0$,
  - $\Phi^{\mathcal{T}}_{gt}(t_{on}) = \Phi^{\mathcal{T}}_{gt}(t_{off}) = exp$ with $\Phi^{\mathcal{T}}_{gt}(t_{random}) = \mathcal{U}_{[1,10]}$,
  - $\Phi^{\mathcal{T}}_{fr}(t_{rain}) = 10, \Phi^{\mathcal{T}}_{fr}(t_{leak}) = 1, \Phi^{\mathcal{T}}_{fr}(t_{irr}) = 20$,
  - $\Phi^{\mathcal{A}}_{w}\langle p_{rainmaker}, t_{rain}\rangle = \Phi^{\mathcal{A}}_{w}\langle p_{rainmaker}, t_{off}\rangle = 2$ and $\Phi^{\mathcal{A}}_{w}(a) = 1$ for all other $a \in \mathcal{A}$, and
  - $\Phi^{\mathcal{T}}_{p}(t) = 1$ for all $t \in \mathcal{T}$.

This HPnG has two discrete fragments $\mathcal{D}_1$ and $\mathcal{D}_2$. The former contains the components controlling the rain, i.e., t_{on}, t_{off}, and $p_{rainmaker}$. The latter contains the components controlling the irrigation, i.e., $p_{control}$, t_{remove}, t_{yes}, t_{no}, p_{choice}, and t_{random}.

Assumptions. We make some assumptions about the structure of the HPnGs considered in this thesis. First, we assume that the HPnGs are finite. Second, we prohibit cycles, which is defined in what follows.

DEFINITION 3.3. **Cyclic HPnGs**

Let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG. We define the underlying directed graph as $G^{dir}_{\mathcal{H}} = (V, E)$ with $V = \mathcal{P} \cup \mathcal{T}$ and $E = \mathcal{A}$. Then, we call $\mathcal{H}$ *cyclic* if $G^{dir}_{\mathcal{H}}$ contains a cycle, and *acyclic* if not.

Let $G^{undir}_{\mathcal{H}}$ be the undirected graph induced by $G^{dir}_{\mathcal{H}}$. Then, we call H *strongly acyclic* if $G^{undir}_{\mathcal{H}}$ does not contain a cycle.

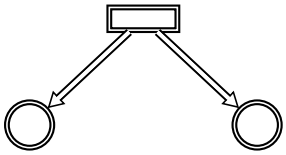


Figure 3.3.: Non-unary HPnG violating the structural requirements.

For example, structures such as cycles of deterministic transitions with a firing time of zero cause Zeno behavior and should be excluded in any case. Moreover, we will later see that the rate adaption algorithm is not guaranteed to terminate in cyclic HPnGs.

Third, we assume that every continuous transition only has one incoming and one outgoing arc to keep the model physically meaningful. HPnGs fulfilling this requirement are called unary. In non-unary structures such as depicted in Figure 3.3, fluid would be duplicated, which physically makes no sense. Also, we will later see that when allowing such structures, our proposed rate adaption algorithm does not terminate.

This restriction is not purely technical. Consider a transition with two outgoing arcs to two different places as depicted in Figure 3.3. There is no semantically equivalent HPnG with only one outgoing arc for the transition. This is because we would have to add at least one transition which is supposed to copy the behavior of the existing transition, in particular, it should fire with the same rate. This cannot be enforced because the syntax of HPnGs does not provide for such a mechanism.

3.2. Semantics of HPnGs

Since we now have an understanding of how HPnGs look like, we can discuss their behavior. In general, the state of a HPnG evolves by moving tokens or fluid between the places. This happens when a transition is firing, which is why we will take a closer look at the firing behavior of the different kinds of transitions. Before doing so, we first define the notion of a state of a HPnG.

DEFINITION 3.4. **State of a HPnG**
A state σ of a HPnG $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ is a tuple (m, x, c, l), where

- $m\colon \mathcal{P}^{disc} \to \mathbb{N}_0$ defines the number of tokens held by each discrete place,
- $x\colon \mathcal{P}^{cont} \to \mathbb{R}_{\geq 0}$ defines the amount of fluid in each continuous place,
- $c\colon \mathcal{T}^{disc} \to \mathbb{R}_{\geq 0}$ defines the firing time (or clock) for discrete transitions,
- $l\colon \mathcal{T}^{cont} \to (\mathcal{P}^{cont} \to [0,1])$ defines the rate restrictions imposed on the continuous transitions by each continuous place.

The set of all states is denoted by $\Sigma^{\mathcal{H}}$. A state σ is *initial* if $\sigma = (m_0, x_0, c, \mathbf{1})$ with $(m_0, x_0) = M_0$ and for $t \in \mathcal{T}^{disc}$, $c(t)$ is either set to $\Phi^{\mathcal{T}}_{ft}(t)$ if t is deterministic or sampled from the distribution $\Phi^{\mathcal{T}}_{gt}(t)$ if t is general. The list of rate restriction $\mathbf{1}$ denotes the function mapping all entries to one, i.e., $l(t)(p) = 1$ for all $t \in \mathcal{T}^{cont}$ and $p \in \mathcal{P}^{cont}$.

As indicated above, each state contains the number of tokens and amount of fluid held in the places. Discrete transitions fire after a specified amount of time. As for SHAs, this is modeled by counting down clock variables. The current clock value of the firing time is also a part of the state. Finally, the state contains a mapping from each continuous transition and each continuous place to a value between zero and one. This is used for the rate adaption procedure and will be discussed in detail in Section 3.2.2. Intuitively, every continuous place can impose restrictions on the adjoining transitions that reduce their firing rate by the given percentage. These restrictions are stored in the restriction list l.

The state is sometimes defined to contain more components, such as the actual firing rates of continuous transitions or their enabling status [Pil+20]. However, as these properties are computable from the components above, we chose to remove redundancies and define the state as compact as possible.

To formally reason about the behavior of the HPnG, we define the notion of *incoming (outgoing) transitions* and *input (output) places*. For each place, the set of incoming transitions contains the transitions connected to the place via an input arc. Dually, the outgoing transitions are those connected via an output arc. For a transition, the intuition for its input and output places is similar.

DEFINITION 3.5. **Input and Output Places and Transitions**

Let $\mathcal{H}$ be a HPnG and $type \in \{disc, cont\}$. For a transition $t \in \mathcal{T}$, we define the *set of input (output) places* as

$$\mathcal{I}^{type}(t) = \{p \mid \langle p, t\rangle \in \mathcal{A}^{type}\} \qquad \mathcal{O}^{type}(t) = \{p \mid \langle t, p\rangle \in \mathcal{A}^{type}\}.$$

Under slight abuse of notation, we define the *set of incoming (outgoing) transitions* for a place $p \in \mathcal{P}$ as

$$\mathcal{I}^{type}(p) = \{t \mid \langle t, p\rangle \in \mathcal{A}^{type}\} \qquad \mathcal{O}^{type}(p) = \{t \mid \langle p, t\rangle \in \mathcal{A}^{type}\}.$$

For $type \in \{test, inh\}$ and a transition $t \in \mathcal{T}$, we define the *set of test (inhibitor) places* as

$$\mathcal{I}^{type}(t) = \{p \mid \langle p, t\rangle \in \mathcal{A}^{type}\}.$$

Notice that the discrete input places of continuous transitions is trivially empty. Dually,

the set of continuous input places of a discrete transition is always empty. Additionally, test and inhibitor arcs are semantically undirected, so we do not need to distinguish between input and output sets. Also, we only consider the input and output places for transitions, and not the incoming and outgoing transitions, as these types of arcs only influence the behavior of transitions and do not affect places. Due to the assumptions made about the arcs of a transition, the set of input and output places is either a singleton or the empty set for every transition.

As already mentioned above, a transition can only fire if it fulfills certain conditions. For example, a discrete transition requires the connected places to hold a specified number of tokens. But even if these requirements are fulfilled, test and inhibitor arcs might prohibit the firing of a transition. We say that a transition has *concession* if all those conditions are met.

DEFINITION 3.6. **Concession**
Let $\mathcal{H}$ be a HPnG. A transition $t \in \mathcal{T}$ has *concession* in a state $\sigma = (m, x, c, l)$, denoted by $conc_\sigma(t)$, iff

1. the discrete places connected to t via a discrete arc contain enough tokens, i.e.,
$$\forall p \in \mathcal{I}^{disc}(t) \colon m(p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle,$$

2. the discrete places connected to t via an inhibitor arc do not contain more tokens than allowed, i.e.,
$$\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{disc}) \colon m(p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle,$$

3. the continuous places connected to t via an inhibitor arc do not contain more fluid than allowed, i.e.,
$$\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{cont}) \colon x(p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle,$$

4. the discrete places connected to t via a test arc do not violate the testing condition, i.e.,
$$\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{disc}) \colon m(p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle, \text{ and}$$

5. the continuous places connected to t via a test arc do not violate the testing condition, i.e.,
$$\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{cont}) \colon x(p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle.$$

Note that conditions (1) and (4) are equal, but as we will discuss in more detail later,

test arcs differ from discrete arcs in their effect on the system as they do not modify the marking. A transition having concession is comparable to a jump being enabled in an SHA. As before, having concession is not necessarily sufficient for a transition to be *fireable*, depending on its type. While continuous transitions are fireable as soon as they have concession, deterministic transitions additionally require that their corresponding clock has run out.

Definition 3.7. Fireable Transitions
Let $\mathcal{H}$ be a HPnG and $\sigma = (m, x, c, l)$ be a state of $\mathcal{H}$. A transition $t \in \mathcal{T}$ is called *fireable*, denoted by $\mathit{fireable}_\sigma(t)$, iff one of the following holds:

- $t \in \mathcal{T}^{disc} \wedge \mathit{conc}_\sigma(t) \wedge c(t) = 0$, or
- $t \in \mathcal{T}^{cont} \wedge \mathit{conc}_\sigma(t)$.

As the name suggests, a transition can fire when it is fireable. Concretely, the firing of a transition means that discrete transitions move tokens and continuous transitions move fluid. If several discrete transitions become fireable at the same time, the arising conflict is solved probabilistically. This will be discussed in more detail in Section 3.2.1. In the literature, a fireable transition is sometimes referred to as *enabled* [GR16; Gha17]. Since this term is already used to describe a property in the SHA semantics that is more similar to having concession in the HPnG semantics, we choose to use the term *fireable* instead. This term has also already been defined for SHAs in Definition 2.9, however, the property described is very similar, so we consider it appropriate to use the same name.

In the optimal case, continuous transitions fire with their nominal firing rate. However, as indicated before, this is not always possible if a place is at its boundary. We therefore compute the *actual firing rate* of a continuous transition by taking into account the current state, more specifically the current list of restrictions.

Definition 3.8. Actual Firing Rate of Continuous Transitions
Let $\mathcal{H}$ be a HPnG and $\sigma = (m, x, c, l) \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$. We define the *actual firing rate* of a continuous transition $t \in \mathcal{T}^{cont}$ as $\mathit{rate}_\sigma \colon \mathcal{T}^{cont} \to \mathbb{R}_{\geq 0}$ with

$$\mathit{rate}_\sigma(t) = \begin{cases} 0 & \text{if } \neg \mathit{fireable}_\sigma(t) \\ \Phi_{fr}^{\mathcal{T}}(t) \cdot \min\left\{ l(t)(p) \;\middle|\; p \in \mathcal{P}^{cont} \right\} & \text{if } \mathit{fireable}_\sigma(t). \end{cases}$$

When a transition is not fireable, it does not fire, meaning that the actual firing rate is zero. If it is fireable, we select the strictest (i.e. smallest) restriction imposed by a continuous place from the restriction list and multiply it with the nominal firing rate. For all continuous transitions t and places p, $l(t)(p)$ gives the percentage of the firing

rate of t that p currently allows. Therefore, the actual firing rate is always smaller than or equal to the nominal firing rate.

We already discussed that the firing of a transition modifies the markings of discrete or continuous places, depending on the type of transition. The following definition formalizes how the firing of a transition affects the current state.

Definition 3.9. Transformed Markings By Firing

Let $\mathcal{H}$ be a HPnG, $t \in \mathcal{T}$ a transition and $\sigma = (m, x, c, l) \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$ such that t is fireable in σ, i.e., $\mathit{fireable}_\sigma(t)$ holds.

If $t \in \mathcal{T}^{disc}$ is a discrete transition, we define the marking resulting from the firing of t as $\mathit{fire}_t(m) \colon \mathcal{P}^{disc} \to \mathbb{N}_0$, where for all $p \in \mathcal{P}^{disc}$

$$\mathit{fire}_t(m)(p) = \begin{cases} m(p) - \Phi^{\mathcal{A}}_w\langle p, t\rangle & \text{if } p \in \mathcal{I}^{disc}(t) \\ m(p) + \Phi^{\mathcal{A}}_w\langle t, p\rangle & \text{if } p \in \mathcal{O}^{disc}(t) \\ m(p) & \text{else.} \end{cases}$$

For a continuous transition $t \in \mathcal{T}^{cont}$ and a time $\tau \in \mathbb{R}_{>0}$, we define the marking resulting from the firing of t for τ time units as $\mathit{fire}^\tau_t(x) \colon \mathcal{P}^{cont} \to \mathbb{R}_{\geq 0}$, where for all $p \in \mathcal{P}^{cont}$

$$\mathit{fire}^\tau_t(x)(p) = \begin{cases} x(p) - \tau \cdot \mathit{rate}_\sigma(t) & \text{if } p \in \mathcal{I}^{cont}(t) \\ x(p) + \tau \cdot \mathit{rate}_\sigma(t) & \text{if } p \in \mathcal{O}^{cont}(t) \\ x(p) & \text{else.} \end{cases}$$

The effect of the firing of multiple continuous transitions $T \subseteq \mathcal{T}^{cont}$ for τ time units is denoted by $\mathit{fire}^\tau_T(x)$, which is formally defined as

$$\mathit{fire}^\tau_T(x)(p) = x(p) + \tau \cdot \sum_{t \in \mathcal{I}^{cont}(p) \cap T} \mathit{rate}_\sigma(t) - \tau \cdot \sum_{t \in \mathcal{O}^{cont}(p) \cap T} \mathit{rate}_\sigma(t).$$

Intuitively, when a discrete transition fires, it moves as many tokens as required by the connecting arcs from the input to the output place. Note that this is well-defined as the transition is only fireable when the input places contain enough tokens.

When a continuous transition fires for τ time units, it moves $\mathit{rate}_\sigma(t) \cdot \tau$ units of fluid from its input to its output place. For now, we are not ensuring that there is enough fluid in the input place and enough capacity in the output place. Therefore, continuous places could under- or overflow. We will present a method to prevent this in Section 3.2.2.

To define the firing behavior of deterministic transitions, we need to make sure that

their assigned clocks are handled correctly. For this, we define two predicates describing how clocks can be modified.

Definition 3.10. Evolution of Clocks

Let $\mathcal{H}$ be a HPnG. For a set of transitions $T \subseteq \mathcal{T}^{disc}$, a clock $c\colon \mathcal{T}^{disc} \to \mathbb{R}_{\geq 0}$ and a time $\tau \in \mathbb{R}_{>0}$, we define the evolution of c as $evolve_T^\tau(c)\colon \mathcal{T}^{disc} \to \mathbb{R}_{\geq 0}$ where for all $t \in \mathcal{T}^{disc}$

$$evolve_T^\tau(c)(t) = \begin{cases} c(t) - \tau & \text{if } t \in T \\ c(t) & \text{else.} \end{cases}$$

Representing the passing of τ time units, $evolve_T^\tau(c)$ decreases the clocks of all transitions in T by τ. When a transition fires, the clocks of this transition are reset depending on the type of transition.

Definition 3.11. Resetting of Clocks

Let $\mathcal{H}$ be a HPnG. For $t \in \mathcal{T}^{disc}$ and $c\colon \mathcal{T}^{disc} \to \mathbb{R}_{\geq 0}$ we define the clock reset by t as $reset_t(c)\colon \mathcal{T}^{disc} \to \mathbb{R}_{\geq 0}$ where for all $t' \in \mathcal{T}^{disc}$

$$reset_t(c)(t') = \begin{cases} c(t') & \text{if } t' \neq t \\ \Phi_{ft}^{\mathcal{T}}(t) & \text{if } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi_{gt}^{\mathcal{T}}(t) & \text{if } t \in \mathcal{T}^{gen}. \end{cases}$$

Intuitively, $reset_t(c)$ updates the clock of a transition t to the firing time $\Phi_{ft}^{\mathcal{T}}(t)$ if t is deterministic, and samples a time from the distribution $\Phi_{gt}^{\mathcal{T}}(t)$ if t is general.

Example 3.2. States of HPnGs

Consider again the HPnG $\mathcal{H}$ from Example 3.1 depicted in Figure 3.2. An initial state of $\mathcal{H}$ is given by $\sigma_0 = (m, x, c, l)$, where

- $m(p_{rainmaker}) = 1, m(p_{control}) = 1, m(p_{choice}) = 0$,
- $x(p_{barr}) = 50, x(p_{plant}) = 0$,
- $c(t_{remove}) = 4, c(t_{yes}) = c(t_{no}) = 0, c(t_{on}) = 1, c(t_{off}) = 2, c(t_{random}) = 5$, and
- $l(t)(p) = 1$ for all $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$.

The state is depicted in Figure 3.4.

The markings m and x are fixed by the initial marking M_0. The clocks for the deterministic transitions are set to the firing time specified by $\Phi_{ft}^{\mathcal{T}}$ and the clocks for the general transitions are sampled from the distribution $\Phi_{gt}^{\mathcal{T}}$. Since no place imposes any restrictions, l is set to 1 for all transitions and places.

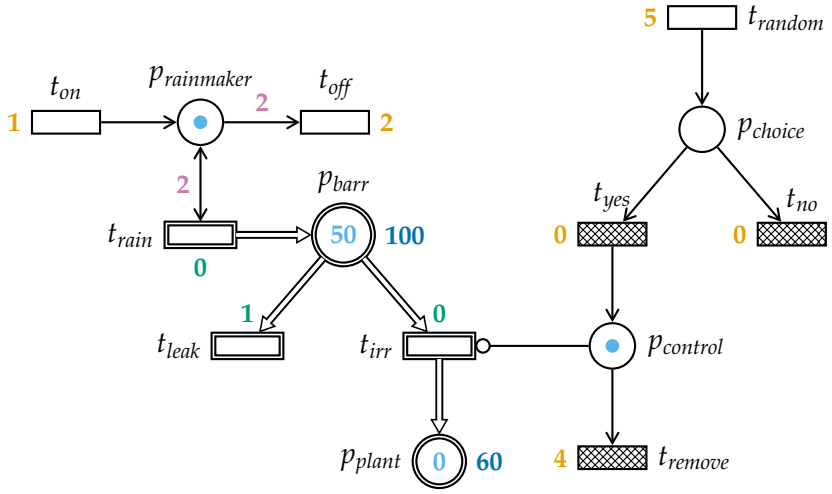


Figure 3.4.: The state of the HPnG $\mathcal{H}$ as discussed in Example 3.2. Instead of firing times and probability distributions, we denote the values of the corresponding clocks next to the discrete transitions. Also, we denote the actual firing rates of transitions instead of the nominal firing rates next to continuous transitions. We still denote the upper bound of the continuous places and the weight of arcs in case they are non-zero.

In σ_0, only one transition is fireable, namely t_{leak}. We have that $rate_{\sigma_0}(t_{leak}) = \Phi^{\mathcal{T}}_{fr}(t_{leak}) = 1$ as there are no restrictions.

The other continuous transitions are not fireable. For example, consider t_{rain}. Since there is a test arc of weight two connecting t_{rain} to $p_{rainmaker}$ requiring $p_{rainmaker}$ to contain at least two tokens, but $m(p_{rainmaker}) = 1$, condition four of Definition 3.6 is violated. Similarly, the token in $p_{control}$ blocks the firing of t_{irr}.

Also, none of the discrete transitions are fireable. Either, their clocks are not zero yet, or, in the case of $c(t_{yes})$ and $c(t_{no})$, the connected place p_{choice} does not contain any tokens.

We consider another state σ_1 depicted in Figure 3.5 with

- $m(p_{rainmaker}) = 2, m(p_{control}) = 0, m(p_{choice}) = 1$
- $x(p_{barr}) = 10, x(p_{plant}) = 10$
- $c(t_{remove}) = 3, c(t_{yes}) = c(t_{no}) = 0, c(t_{on}) = 3, c(t_{off}) = 0, c(t_{random}) = 6,$
- $l(t)(p) = 1$ for all $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$.

This is not an initial state, for example, because the marking is not the initial marking. However, we can see that several transitions are fireable: There are two tokens in $m(p_{rainmaker})$, so t_{rain} is fireable. Since there are still no restrictions, the actual firing rate of t_{rain} is equal to the nominal firing rate and we have

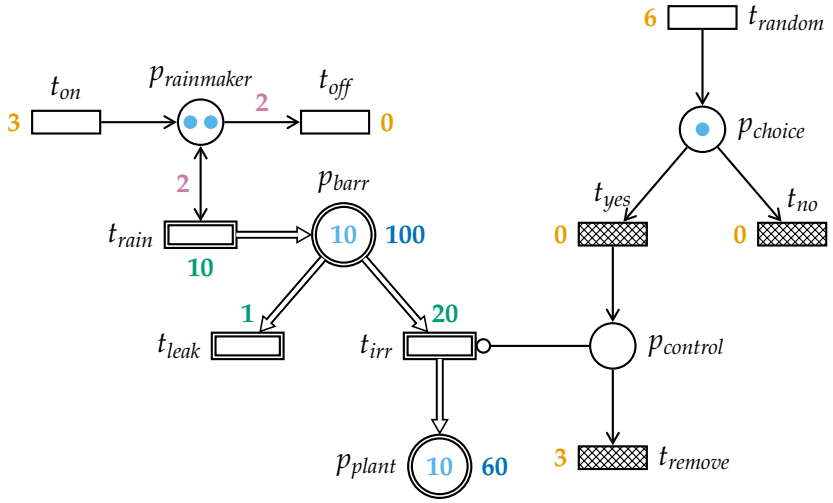


Figure 3.5.: The second state of the HPnG $\mathcal{H}$ discussed in Example 3.2.

$rate_{\sigma_1}(t_{rain}) = \Phi^{\mathcal{T}}_{fr}(t_{rain}) = 10$. Transition t_{irr} is also fireable and fires with its nominal firing rate because $m(p_{control}) = 0$, and the same applies to t_{leak}.

The discrete transitions $c(t_{yes})$ and $c(t_{no})$ are also fireable, because p_{choice} contains a token. Only one of them can fire though, since firing either transition will remove the token from the place. In the next section, we discuss how to handle such conflicts.

We will use the predicates defined in the preceding part to define inference rules in Section 3.2.3. Before we can do so, we discuss the handling of conflicts for deterministic transitions in Section 3.2.1 and for continuous transitions in Section 3.2.2.

3.2.1. Conflict Resolution for Deterministic Transitions

In general, we can solve conflicts arising between discrete transitions either by non-deterministically choosing a transition that is allowed to fire or by assigning firing probabilities to all competing transitions. We will use the latter approach and assign probabilities to transitions proportional to their priority. More precisely, for a transition $t \in \mathcal{T}^{disc}$ fireable in a state σ, we take t with the following probability:

$$\frac{\Phi^{\mathcal{T}}_p(t)}{\sum_{t' \in \mathcal{T}^{disc},\, fireable_\sigma(t')} \Phi^{\mathcal{T}}_p(t')}.$$

Deterministic transitions can be in direct conflict, for example, if they are competing for a token as illustrated in Example 3.2. However, the order of firing can also change

the resulting state in other scenarios, for example if a transition moves tokens to a place which then inhibits the firing of another transition. Consequently, we consider *all* fireable discrete transitions, as opposed to only those emanating from the same place.

General transitions can theoretically also be in conflict with other transitions. However, since the probability distributions that determine the firing times are continuous, the probability of this happening is zero. Therefore, we do not consider these cases here.

EXAMPLE 3.3. **Conflicting Deterministic Transitions**

Consider again the state $\sigma_1 = (m, x, c, l)$ presented in Example 3.2 and depicted in Figure 3.5 with

- $m(p_{rainmaker}) = 2, m(p_{control}) = 0, m(p_{choice}) = 1,$
- $x(p_{barr}) = 10, x(p_{plant}) = 10,$
- $c(t_{remove}) = 3, c(t_{yes}) = c(t_{no}) = 0, c(t_{on}) = 3, c(t_{off}) = 0, c(t_{random}) = 6,$ and
- $l(t)(p) = 1$ for all $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$.

As was already mentioned in Example 3.2, both t_{yes} and t_{no} are fireable. We have that $\Phi_p^{\mathcal{T}}(t_{yes}) = \Phi_p^{\mathcal{T}}(t_{no}) = 1$, so this conflict is solved by choosing either transition with probability $\frac{1}{2}$.

3.2.2. Conflict Resolution for Continuous Transitions

As mentioned above, fluid transitions can cause conflicts when trying to move fluid out of an empty place or into a full place. We refer to the difference between inflow and outflow of a continuous place as the *drift*.

DEFINITION 3.12. **Drift of a Continuous Place**

Let $\mathcal{H}$ be a HPnG and $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$. We define the *drift* of the continuous places $\mathcal{P}^{cont}$ in σ as $drift_\sigma \colon \mathcal{P}^{cont} \to \mathbb{R}$, where

$$drift_\sigma(p) = \sum_{t_i \in \mathcal{I}^{cont}(p)} rate_\sigma(t_i) - \sum_{t_j \in \mathcal{O}^{cont}(p)} rate_\sigma(t_j)$$

for $p \in \mathcal{P}^{cont}$.

Notice that using this definition, we can give a simplified version of the continuous marking resulting from the firing of continuous transitions in Definition 3.9. Let T be

the set of fireable transitions in σ. Then, for $p \in \mathcal{P}^{cont}$,

$$\begin{aligned} fire_T^\tau(x)(p) &= x(p) + \tau \cdot \sum_{t \in \mathcal{I}^{cont}(p) \cap T} rate_\sigma(t) - \tau \cdot \sum_{t \in \mathcal{O}^{cont}(p) \cap T} rate_\sigma(t) \\ &= x(p) + \tau \cdot drift_\sigma(p). \end{aligned}$$

A place can neither hold a negative amount of fluid nor more than its specified capacity. Therefore, we need to adapt the current firing rates of fireable transitions in a state $\sigma = (m, x, c, l)$ if a place $p \in \mathcal{P}^{cont}$ is at its boundary and the drift leads the place to under- or overflow, i.e., $x(p) = 0$ and $drift_\sigma(p) < 0$ or $x(p) = \Phi_{ub}^{\mathcal{P}}(p)$ and $drift_\sigma(p) > 0$. We say that the drift of an empty (full) place goes in the *wrong direction* if it is negative (positive), otherwise is goes in the *right direction*. Places that are either at upper or at lower boundary are referred to as *critical* places.

We propose a novel algorithm for rate adaption based on two principles: First, we aim to reduce the firing rates of transitions proportional to the current firing rate. Second, we add a policy of mutual exclusion. While places are always allowed to further reduce the rate, we restrict the revoking of reductions. A place is allowed to revoke its own restrictions, but has to respect the reductions made by other places. A detailed comparison to existing algorithms is given in Section 3.2.2.

We distinguish four cases, namely rate reduction and rate reset for places at lower boundary and the dual cases for places at upper boundary. Consider a place $p \in \mathcal{P}^{cont}$ at lower boundary in a state σ. If the drift goes in the wrong direction (i.e. is negative), we reduce all outgoing transitions proportionally. If the drift goes in the right direction (i.e. is positive), we revoke all restrictions of the outgoing transitions made by p. This might change the current rates, but does not have to, as other places might demand the same or even stronger restrictions. For places at upper boundary, the steps are dual. In what follows, we provide a formal description of the full algorithm.

The Rate Adaption Algorithm

Let $\mathcal{H}$ be a HPnG, $\sigma = (m, x, c, l) \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $p \in \mathcal{P}^{cont}$ a continuous place. For the first case, namely the *rate reduction*, assume that p is a critical place and has drift in the wrong direction. We then adapt the outgoing, respectively the incoming, transitions of p in such a way that the drift of p becomes zero.

> Definition 3.13. **Rate Reduction**
> Let $\mathcal{H}$ be a HPnG, $\sigma = (m, x, c, l) \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $p \in \mathcal{P}^{cont}$ a continuous place. Further, let
> $$in_\sigma(p) = \sum_{t \in \mathcal{I}^{cont}(p)} rate_\sigma(t)$$

be the incoming flow to place p and

$$out_\sigma(p) = \sum_{t \in \mathcal{O}^{cont}(p)} rate_\sigma(t)$$

the outgoing flow to place p.

If $x(p) = 0$ and $drift_\sigma(p) < 0$, we let $T = \left\{t \in \mathcal{O}^{cont}(p) \;\middle|\; fireable_\sigma(t)\right\}$ be the outgoing fireable transitions of p and define

$$reduce_p^{empty}(l) \colon \mathcal{T}^{cont} \to (\mathcal{P}^{cont} \to [0,1])$$

as

$$reduce_p^{empty}(l)(t)(p') = \begin{cases} \frac{in_\sigma(p)}{out_\sigma(p)} \cdot \overbrace{\min\left\{l(t)(p'') \;\middle|\; p'' \in \mathcal{P}^{cont}\right\}}^{\text{current strongest restriction}} & \text{if } t \in T \wedge p = p' \\ l(t)(p') & \text{else} \end{cases}$$

for all $p' \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont}$.

Dually, if $x(p) = \Phi_{ub}^{\mathcal{P}}(p)$ and $drift_\sigma(p) > 0$, we let $T = \left\{t \in \mathcal{I}^{cont}(p) \;\middle|\; fireable_\sigma(t)\right\}$ be the incoming fireable transitions of p and define

$$reduce_p^{full}(l) \colon \mathcal{T}^{cont} \to (\mathcal{P}^{cont} \to [0,1])$$

as

$$reduce_p^{full}(l)(t)(p') = \begin{cases} \frac{out_\sigma(p)}{in_\sigma(p)} \cdot \overbrace{\min\left\{l(t)(p'') \;\middle|\; p'' \in \mathcal{P}^{cont}\right\}}^{\text{current strongest restriction}} & \text{if } t \in T \wedge p = p' \\ l(t)(p') & \text{else} \end{cases}$$

for all $p' \in \mathcal{P}^{cont}$ and $t \in \mathcal{T}^{cont}$.

Thus, if p is empty, we distribute the incoming flow $in(p)$ to the outgoing fireable transitions proportional to their actual firing rate. As the actual firing rate is computed from the list of restrictions, this is equal to setting the new restriction for a transition t to the current strongest restriction scaled by $\frac{in(p)}{out(p)}$. Since we know that the drift is negative, we have that $in(p) < out(p)$. Consequently, the fraction $\frac{in(p)}{out(p)}$ is strictly smaller than one and the updated restriction must be stronger than the previously strongest restriction.

Furthermore, it can be easily verified that, after the reduction, the outgoing flow matches the incoming flow: The outgoing flow for p in the new state $\sigma' = (m, x, c, l')$

with $l' = reduce_p^{empty}(l)$ can be computed as

$$\begin{aligned}
&\sum_{t \in \mathcal{O}^{cont}(p)} rate_{\sigma'}(t) \\
&= \sum_{t \in T} rate_{\sigma'}(t) \\
&= \sum_{t \in T} \left(\min \left\{ l'(t)(p') \mid p' \in \mathcal{P}^{cont} \right\} \cdot \Phi_{fr}^{\mathcal{T}}(t) \right) \\
&= \sum_{t \in T} \left(\frac{in(p)}{out(p)} \cdot \min \left\{ l(t)(p') \mid p' \in \mathcal{P}^{cont} \right\} \cdot \Phi_{fr}^{\mathcal{T}}(t) \right) \\
&= \frac{in(p)}{out(p)} \cdot \sum_{t \in T} \left(\min \left\{ l(t)(p') \mid p' \in \mathcal{P}^{cont} \right\} \cdot \Phi_{fr}^{\mathcal{T}}(t) \right) \\
&= \frac{in(p)}{out(p)} \cdot \sum_{t \in T} rate_{\sigma}(t) \\
&= \frac{in(p)}{out(p)} \cdot out(p) \\
&= in(p),
\end{aligned}$$

as desired. The dual statement holds for $reduce_p^{full}(l)$ for places at upper boundary.

If the upper bound equals zero, i.e., $\Phi_{ub}^{\mathcal{P}}(p) = 0$ for a place p, we can reduce either the outgoing or the incoming transitions to obtain a drift of zero. In such cases, we always reduce incoming transitions.

This concludes the first case of the algorithm, the rate reduction. Now, we consider the second case, so assume place p is critical, but with a non-zero drift in the right direction. We then remove the restrictions imposed by p on the outgoing transitions. This step is solely taken if p previously imposed restrictions on transitions that have not been revoked yet, so if there exists a transition $t \in \mathcal{T}^{cont}$ such that $l(t)(p) < 1$. This is expressed in the following predicate.

> Definition 3.14. **Placing Restrictions**
>
> Let $\mathcal{H}$ be a HPnG, $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $p \in \mathcal{P}^{cont}$ a continuous place. We denote by $restr_\sigma(p)$ whether p imposes restrictions on some transition in σ, i.e., $restr_\sigma(p)$ is true iff there exists a transition $t \in \mathcal{I}^{cont}(p) \cup \mathcal{O}^{cont}(p)$ such that $l(t)(p) < 1$.

Now, we define the rate reset by simply setting all entries in the restriction list belonging to p to one. Then, p no longer influences the firing rates of any of its outgoing or incoming transitions.

Definition 3.15. **Rate Reset**

Let $\mathcal{H}$ be a HPnG, $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $p \in \mathcal{P}^{cont}$ a continuous place. We define the updated restriction list $reset_p(l) \colon \mathcal{T}^{cont} \to (\mathcal{P}^{cont} \to [0,1])$ as

$$reset_p(l)(t)(p') = \begin{cases} 1 & \text{if } p = p' \\ l(t)(p') & \text{else} \end{cases}$$

for each $p' \in \mathcal{P}^{cont}$ and $t \in \mathcal{T}^{cont}$.

The rate reduction and rate reset steps form the basis of the rate adaption algorithm. Since adapting the rates at one place might make it necessary to adapt rates at another place, we need to apply the steps until no further steps are possible and all conflicts are solved. The full rate adaption algorithm is given in Algorithm 1.

In every iteration of the while-loop, we check if any place p requires a rate reduction or a rate reset and calculate the new restriction list l' if necessary. Note that the third and fourth case have the same effect; we split the cases here for better readability. The *continue* flag makes sure that the while-loop is continued as long as there exists a critical place with drift in the wrong direction or a critical place with drift in the right direction that imposes restrictions. This corresponds to one of the if-conditions being true. When all conflicts are solved, the algorithm returns the new state σ' with the updated list of restrictions.

The algorithm terminates when the drift is in the right direction for every critical place. Note that we do not require all critical places to have a drift of zero after the algorithm terminates. Such a solution does not always exist (see Example 3.7 for an example). Nevertheless, the resulting state is still stable, in the sense that rates are not changed back and forth. This is due to the fact that the algorithm only attempts to reset the rates at a place if this place still sets restrictions.

When a place p restricts a transition t that is then further reduced by another place, we fully reset the rates before potentially reducing again. This could also be handled in a separate step that adjusts the rates when it is necessary to relax the restrictions a little, but not completely. As an advantage, one would save a step and prevent that a place resets rates too optimistically, making the algorithm more efficient. However, since the effects of one adjustment are the same as first resetting and then reducing again, we choose not to explicitly distinguish the former. We compare this algorithm to existing algorithms in Section 3.2.2 after presenting an example and discussing its termination.

Input: HPnG $\mathcal{H}$, state $\sigma = (x, m, c, l) \in \Sigma^{\mathcal{H}}$
Output: State $\sigma' \in \Sigma^{\mathcal{H}}$
$continue = true$;
$\sigma' = \sigma$;
while *continue* **do**
  for $p \in \mathcal{P}^{cont}$ *critical* **do**
    if $x(p) = 0$ *and* $drift_{\sigma'}(p) < 0$ **then**
      $l' = reduce_p^{empty}(l')$;
      $continue = true$;
    else if $x(p) = \Phi_{ub}^{\mathcal{P}}(p)$ *and* $drift_{\sigma'}(p) > 0$ **then**
      $l' = reduce_p^{full}(l')$;
      $continue = true$;
    else if $x(p) = 0$ *and* $drift_{\sigma'}(p) < 0$ *and* $restr_{\sigma'}(p)$ **then**
      $l' = reset_p(l')$;
      $continue = true$;
    else if $x(p) = \Phi_{ub}^{\mathcal{P}}(p)$ *and* $drift_{\sigma'}(p) > 0$ *and* $restr_{\sigma'}(p)$ **then**
      $l' = reset_p(l')$;
      $continue = true$;
    else
      $continue = false$;
    end
    $\sigma' = (x, m, c, l')$;
  end
end
return σ'

Algorithm 1: The rate adaption algorithm.

EXAMPLE 3.4. **Rate Adaption**

We consider the state σ of the HPnG $\mathcal{H}$ presented in Example 3.1. The state is depicted in Figure 3.6 and is formally given by $\sigma = (m, x, c, l)$ with

- $m(p_{rainmaker}) = 2, m(p_{control}) = 0, m(p_{choice}) = 0$,
- $x(p_{barr}) = 80, x(p_{plant}) = 60$,
- $c(t_{remove}) = 3, c(t_{yes}) = c(t_{no}) = 0, c(t_{on}) = 3, c(t_{off}) = 4, c(t_{random}) = 6$, and
- $l(t)(p) = 1$ for all $t \in \mathcal{T}^{cont}$ and $p \in \mathcal{P}^{cont}$.

The place p_{plant} is at its upper boundary $\Phi_{ub}^{\mathcal{P}}(p_{plant}) = 60$. However, the transition t_{irr} still fires at its full rate $\Phi_{fr}^{\mathcal{T}}(t_{irr}) = 20$, causing the plants to be overwatered. The rate adaption algorithm solves this by computing an updated list of restrictions

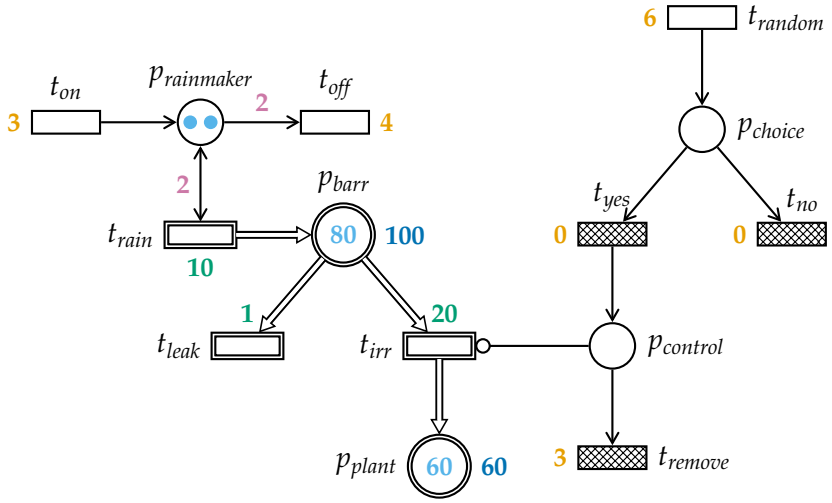


Figure 3.6.: The state of the HPnG $\mathcal{H}$ discussed in Example 3.4.

$reduce^{full}_{p_{plant}}(l)$, which is zero at the entry for t_{irr} and p_{plant}. Consequently, the actual firing rate of t_{irr} in the new state is zero.

Note that the barrel is almost full as well. Since the irrigation no longer takes water from the place, but it still rains, the barrel fills up. We might reach a state σ' as depicted in Figure 3.7 and formally given by

- $m(p_{rainmaker}) = 2, m(p_{control}) = 0, m(p_{choice}) = 0$,
- $x(p_{barr}) = 100, x(p_{plant}) = 60$,
- $c(t_{remove}) = 3, c(t_{yes}) = c(t_{no}) = 0, c(t_{on}) = 3, c(t_{off}) = 4, c(t_{random}) = 6$, and
- $l(t_{irr})(p_{plant}) = 0$, $l(t)(p) = 1$ for all other $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$.

Then, we update the list of restrictions at the entry for t_{rain} and p_{barr} by

$$reduce^{full}_{p_{barr}}(l)(t_{rain})(p_{barr}) = \frac{out(p_{barr})}{in(p_{barr})} \cdot \overbrace{\min\left\{l(t_{rain})(p'') \mid p'' \in \mathcal{P}^{cont}\right\}}^{\text{current strongest restriction}}$$
$$= \frac{1}{10} \cdot 1 = 0.1$$

and hence have that the actual firing rate of t_{rain} in the new state is $0.1 \cdot 10 = 1$.

Should the rain stop at some later point in time, the drift of p_{barr} is in the right direction again. Then, $reset_{p_{barr}}(l)$ resets the respective restriction list entry to one, effectively removing the restriction imposed by p_{barr}.

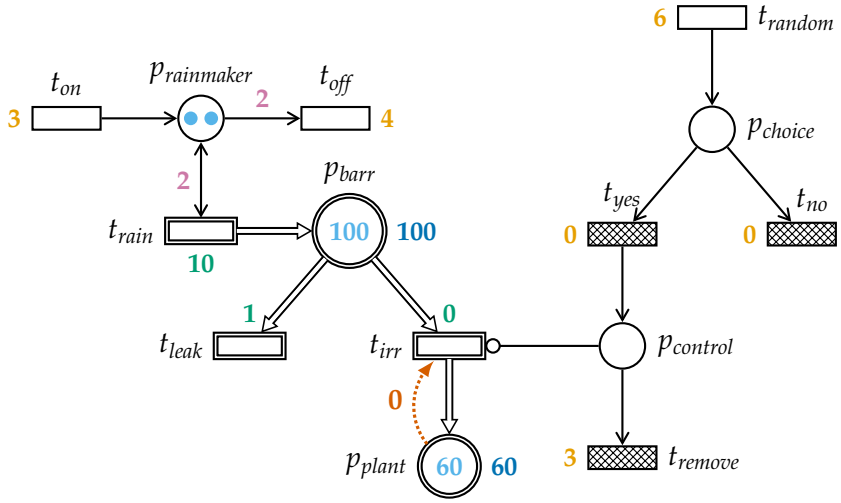


Figure 3.7.: The second state of the HPnG $\mathcal{H}$ discussed in Example 3.4. The restriction placed on t_{irr} by p_{plant} is denoted by a dashed arrow.

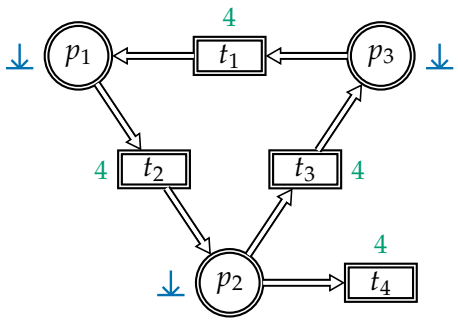


Figure 3.8.: A cyclic HPnG with non-terminating rate adaption as discussed in Example 3.5. We denote by ⇓ when a place is empty and write the firing rates of the continuous transition beside it.

Termination of the Rate Adaption Algorithm

As described above, several places may be in need of rate adaption at the same time. Then, Algorithm 1 adapts the rates for every affected place. However, this might cause the drift of another critical place to go in the wrong direction. Then, we have to adapt the rates at these places as well. It is not obvious that this always terminates. To illustrate this, we start by giving some examples of HPnGs with weaker assumptions and show that the rate adaption algorithm does not terminate. Recall that as described in Section 3.1, we require HPnGs to be acyclic and assume that every transition only has one incoming and one outgoing arc.

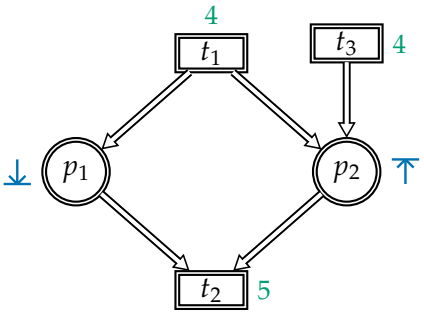


Figure 3.9.: HPnG violating the assumptions with non-terminating rate adaption as discussed in Example 3.6. Again, we denote an empty place by ⤓ and a full place by ⤒.

EXAMPLE 3.5. **Non-Terminating Rate Adaption in Cyclic HPnGs**

Consider the cyclic HPnG shown in Figure 3.8. All places are empty and all transitions have a current rate of four. Place p_2 has a negative drift of four, because the incoming flow is four and the outgoing flow is eight, and therefore has to adapt the rates.

In the first step, p_2 sets the rates of t_3 and t_4 to two. Both p_3 and p_1 react by setting the rates of t_1 and t_2 to two as well.

Then, p_2 has to adapt again and halves the rates of its outgoing transitions. It can already be seen that this will not terminate, as p_2 will continue to halve the rates infinitely often.

It is not possible to prevent infinite adaptions as in Example 3.5 when allowing cycles, hence our restriction to acyclic HPnGs. However, non-termination can also occur in the absence of cycles. The next example shows a HPnG that, contrary to our assumption, contains transitions with multiple incoming and outgoing arcs.

EXAMPLE 3.6. **Non-Terminating Rate Adaption for Transitions with Multiple Arcs**

The HPnG depicted in Figure 3.9 violates our assumptions, as transition t_1 has two outgoing arcs and t_2 has two incoming arcs. Both places p_1 and p_2 are critical and have drift in the wrong direction. Therefore, p_1 sets the rate of t_2 to four, matching its incoming flow. Next, p_2 reduces the rate of both incoming transitions t_1 and t_3 to two. Then, the drift at p_1 is negative again, so the rate of t_2 is reduced to two as well. For the same reasons as in Example 3.5, this will not terminate.

Examples 3.5 and 3.6 showed that, if we disregard our assumptions, Algorithm 1 does not necessarily terminate. Now, we return to assuming that our restrictions hold. To get a better understanding of the termination behavior, we discuss another example.

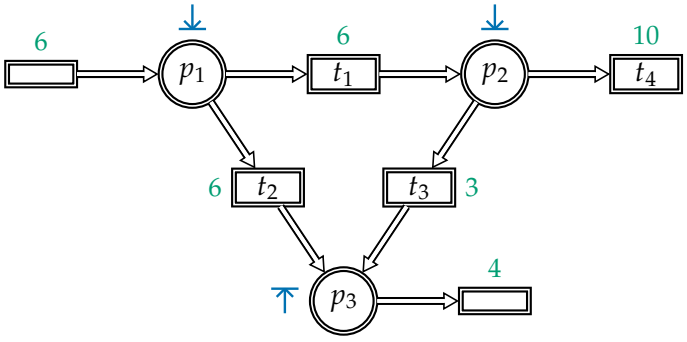


Figure 3.10.: Acyclic but non-strongly acyclic HPnG discussed in Example 3.7.

EXAMPLE 3.7. **Termination of (non-strongly) acyclic HPnGs**

Consider the HPnG in Figure 3.10, which is acyclic but not strongly acyclic. For this HPnG, all assumptions specified in Section 3.1 are fulfilled. We present a possible execution of the rate adaption algorithm in the following table. Note that we disregard the firing rates of the two unnamed source and sink transitions, as they are not modified. We denote the step taken in the action column by $\downarrow p$ if a place p reduces the rates of its outgoing or incoming transitions and $\uparrow p$ if p resets the rates. Choosing the actions in a different order can change the result of the algorithm, but not the termination.

In the first step, p_1 reduces both outgoing transitions by 50%. Then, p_3 reduces its incoming transitions t_2 and t_3 in step two. Note that this is done proportionally to the current firing rates, not the nominal rates, which is why both transitions are assigned the same new rate even though their nominal rates differ. Transition t_2 is now reduced stronger than required by p_1, so all restrictions made by p_1 are revoked in step three. As a consequence, the rate of t_1 is reset to its nominal rate. Place p_1 now has a negative drift again, because the nominal rate is too high.

Step	Action	t_1	t_2	t_3	t_4
0	-	6	6	3	10
1	$\downarrow p_1$	3	3	3	10
2	$\downarrow p_3$	3	2	2	10
3	$\uparrow p_1$	6	2	2	10
4	$\downarrow p_2$	6	2	1	5
5	$\downarrow p_1$	4.5	1.5	1	5
6	$\downarrow p_2$	4.5	1.5	0.75	3.75
7	$\uparrow p_3$	4.5	1.5	0.75	3.75

Next, p_2 reduces its outgoing transitions in step four. Then, p_1 again throttles the rate of t_1 in step five and p_2 has to adapt the rates in the next step. Now, the drift of p_3 is negative, which is the right direction as p_3 is full, and the place tries to reset the rates of the incoming transitions in step seven. However, since p_1 caused the current strongest restriction on t_1 and p_2 caused the current restriction on t_3, p_3

cannot revoke the reductions. Therefore, all places have a drift in the right direction and the algorithm terminates.

As can be seen, the algorithm does not terminate in a straightforward manner, but considers places several times. We did not yet find an upper bound on these visits. Nevertheless, we were not able to construct an example that respects the restrictions and still does not terminate. This suggests that the rate adaption does terminate, as formulated in Conjecture 3.1.

CONJECTURE 3.1. **Termination of Rate Adaption**

Let $\mathcal{H}$ be a acyclic unary HPnG and $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$. Then, Algorithm 1 applied to $\mathcal{H}$ and σ terminates.

This has not yet been proven for the full class of HPnGs. As a silver lining, we prove termination for specific states of a HPnG and for the subclass of strongly acyclic HPnGs.

THEOREM 3.1. **Partial Termination of Rate Adaption**

Let $\mathcal{H}$ be a HPnG and $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$. Then, Algorithm 1 applied to $\mathcal{H}$ and σ terminates

1. if in σ, either all critical places are empty or all critical places are full, or
2. if $\mathcal{H}$ is strongly acyclic.

Proof. Let $\mathcal{H}$ be a HPnG and $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$.

Part One. Assume that in σ no place is at its upper boundary and thus all critical places are empty. We assume that $\mathcal{H}$ is ordered such that source transitions are on the top and subsequent transitions are below. Then, we say that a transition t_1 is *above* another transition t_2, denoted by $t_1 > t_2$, if it is closer to the source. Formally, this is given if $t_1 \neq t_2$ and there exists a path in the underlying directed graph from a source transition to t_1 and from t_1 to t_2.

First, note that since we only consider empty places in σ and transitions are assumed to only have at most one incoming and one outgoing place, each transition in this scenario is controlled by at most one place. Additionally, each place can adjust the rates only once without occasion, where an occasion is given if the incoming transitions change their rates. Each subsequent adjustment is a reaction to adjustments from above.

For places that are only connected to source transitions, i.e., transitions without incoming places, this means that they will only adapt their rates once. All other places readapt their rates (if necessary) when the incoming transitions change their rates. This only happens when the input place of these transitions adapts its rates. We refer to

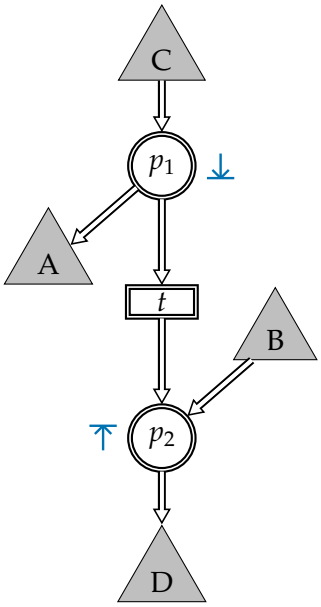


Figure 3.11.: Illustration of a HPnG where a transition is reduced by two places, one empty (p_1) and one full (p_2), causing non-straightforward termination as discussed in the second part of the proof for Theorem 3.1. Parts of the HPnG we do not consider in more detail here are depicted in the four triangles A,B,C, and D.

those places as predecessor places. Therefore, the maximal number of adaptations per place is given by one plus the maximal number of adaptations per predecessor place.

There can be no loops in this propagation, as we exclude cyclic structures such as Figure 3.8 per assumption and each place only causes the places below them to readapt. Since we only consider finite HPnGs, there can neither be an infinite number of predecessors nor an infinite chain of connected places. Therefore, the algorithm terminates after a finite number of steps.

This proof can be carried out in the same way if only places at the upper boundary occur. The only difference is that full places are adapting the rates of their incoming transitions, so adaptions are propagated upwards instead of downwards.

Part Two. We now discuss the termination of the algorithm in a state that contains both empty and full places. For this, we assume $\mathcal{H}$ to be strongly acyclic.

First, note that the only interesting case is when there is a transition connecting an empty place to a full place, because only then, both are potentially imposing restrictions on the same transition. This is visualized in Figure 3.11. If there are non-critical places in between an empty and a full place, then this does not affect the termination at all, because the drift of the non-critical place can absorb everything and does not propagate any adaptions. If a full place is above an empty place, they also do not affect each

other because they propagate their adaptions in opposite directions and the transition connecting them will never change the rate.

Therefore, this part differs from the first part of the proof if the HPnG contains a transition t controlled by two places as depicted in Figure 3.11. Since p_1 is empty, it has to reduce the rate of t, and dually p_2 is full and also reduces the rate of t. Let us for the moment assume that A, B, C, and D each only contain connected places at the same kind of boundary, so t is the only transition controlled by two places. We then know from the first part of the proof that the algorithm terminates for those subnets in finite time, if p_1 and p_2 only give them finitely many reasons to adapt. Also, since the HPnG is assumed to be strongly acyclic, we know that there is no connection between the subnets. Consequently, they cannot influence each other.

We now show that places p_1 and p_2 are only finitely often adapting and resetting the rates of their connecting transition t. We do so by arguing about the number of times p_1 and p_2 adapt their rates. As we already mentioned above, places are only readapting their rates if their neighboring places are adapting. Therefore, if we can show that the neighboring places are only finitely often adapting their rates, we can conclude that the place itself also adapts finitely often.

Again, we know that the calculation terminates for subnets C and D, so they give only finitely many reasons to adapt. Without loss of generality, we assume that p_1 adapts the rate of t and its other outgoing transitions in A before p_2. Using the first part, we know that the initiated calculation of rate adaption in A terminates.

At some later point in the algorithm, p_2 adapts the rates. By definition, p_2 restricts t stronger than p_1 did. Then, p_1 resets and eventually readapts the rates in A; both causing finitely many adaptions in A. Further adaptions coming from C or D can also only cause finitely many adaptions and resets. Therefore, the algorithm terminates for HPnGs structured as in Figure 3.11.

By induction on the number of meeting points of empty and full places, we can conclude that the algorithm also terminates if there is an arbitrary number of such structures. Using the same arguments as above, we know that there are always only finitely many adaptions initiated by the neighboring places. We can conclude that the algorithm terminates for strongly acyclic HPnGs. □

It remains as future work to extend this proof to (non-strongly) acyclic HPnGs. Example 3.7 illustrated the challenges in characterizing the termination of HPnGs that are not strongly acyclic. The proof given above fails at the point where we assume that parts A, B, C, and D are not connected.

Ideas for proving the termination include, for example, providing an upper bound of the number of visits to a place. Also, if we could define a well-founded order over the

states of a HPnG and show that applying the rate adaption decreases the order, we can conclude that there are no infinite descending chains and thus the algorithm terminates. Ideas for such an order were, for example, to express the degree of the violation in form of the size of the drift in the wrong direction. One could then incorporate the way that the drifts in the wrong direction propagate through the HPnG: For places at lower boundary, the problem shifts further away from the source transitions with each iteration of rate adaption. We were not yet able to define a suitable ordering.

Another promising idea is to maintain tree-like structures that represent which place has caused the necessity of (re-)adaption for which places. The roots of the trees are the critical places with drift in the wrong direction in the initial state. Then, if a place p causes another place p' to adapt the rates, p' is added as a child of p under the condition that p' is not already a leaf in any of the trees. Thus, a forest is gradually formed during the execution of the algorithm. If one can prove that the depth of each tree is finite, termination of the algorithm is implied.

Comparison to Other Rate Adaption Algorithms

Our rate adaption algorithm given in Algorithm 1 can be characterized by two main properties: First, we adapt transitions proportionally to their actual firing rate, formally described in Definition 3.13. Second, we only let places increase firing rates as long as no other place sets stronger restrictions. This is implemented in the semantics by maintaining a list of restrictions for every continuous transition and defining the actual firing rate with respect to the strongest restriction (see Definition 3.8). We now briefly present other possible rate adaption algorithms.

In contrast to adapting the rates proportionally to the actual firing rate, we could adapt proportionally to the nominal firing rate. This is a bit more intricate, as we then actively have to make sure that we respect the restrictions set by other places. While both approaches are valid, we decided for an adaption proportional to the actual rate, as it is both simpler and closer to the physical processes we want to model.

A method originally proposed by David and Alla uses priorities to solve conflicts of continuous places [DA10]. Each transition is assigned a priority, which has to be unique with respect to its input and output place, i.e., the priorities of two transitions originating from or leading to the same place have to differ. Then, a transition is only assigned a positive flow rate if all transitions with higher priority can fire at their nominal rate. For details, we refer to [DA10, Section 5.3.2] and [Gha17, Section 3.5.1].

It is quite tedious to make sure that priorities are unique in the way described above. Therefore, an extension of the previous method additionally assigns a share to each transition which is used to distribute the fluid in case two transitions have the same priority. Again, we refer to [DA10, Section 5.3.3] and [Gha17, Section 3.5.2] for details.

All methods for a modified rate update can be straightforwardly included in our framework. Shares and priorities can be added as parameter functions. Then, the function describing the rate reduction from Definition 3.13 can be replaced by a function implementing the respective reduction method instead.

Our algorithm terminates for all examples presented by Ghasemieh, in particular for those where the other algorithms do not terminate [Gha17, Figures 3.13 and 3.14]. In many cases, this is attributable to the blocking of rate resets, i.e., the fact that in our model, a place can only reverse its own restrictions. There is no comparable construct in the other algorithms.

The termination of algorithms distributing fluid using priorities or shares is only guaranteed under the assumption of two hypotheses originally formulated by David and Alla [DA10]. The first hypothesis expresses that every transition is only allowed to be involved in one conflict. This already excludes all conflicts that are structured as outlined in Figure 3.11. Therefore, this restriction is at least as strong as our restriction to strongly acyclic HPnGs. The second hypothesis states that there is no feedback allowed, i.e., the result of solving one conflict may not influence another conflict. Thus, these restrictions are quite strict. In particular, assuming both hypotheses, our rate adaption algorithm terminates. This does not hold in the other direction: Our algorithm terminates for some examples while the others do not.

In conclusion, the rate adaption algorithm proposed in this thesis implements a conceptually simple method for reducing rates. At the same time, we can extend the algorithm easily to include more involved strategies.´ Additionally, we mitigate termination requirements by preventing places from lifting restrictions they did not cause.

3.2.3. Operational Semantics

The findings of the previous parts of this section are now summarized in seven inference rules that describe how a state of a HPnG evolves. We again let $\mathcal{H}$ be a HPnG and $\sigma = (m, x, c, l) \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$. We distinguish between state changes initiated by the firing of deterministic transitions, by rate adaption and by the passing of time.

Firing of Deterministic Transitions. A deterministic transition $t \in \mathcal{T}^{disc}$ can fire as soon as it becomes fireable. The effect on σ is that the discrete marking of the connected places changes and the clock of the transition is reset. This is summarized in the following rule for $\sigma = (m, x, c, l)$:

$$\frac{t \in \mathcal{T}^{disc} \quad fireable_{\sigma}(t)}{(m, x, c, l) \xrightarrow{\text{fire } t} (fire_t(m), x, reset_t(c), l)} \text{ fire}$$

If several such rules are applicable at the same time, i.e., deterministic transitions are in conflict, we solve this as described in Section 3.2.1.

Rate Adaption. The rate adaption algorithm defined in Algorithm 1 can be translated for $\sigma = (m, x, c, l)$:

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = 0 \quad drift_\sigma(p) < 0}{(m, x, c, l) \xRightarrow{\text{reduce empty } p} (m, x, c, reduce_p^{empty}(l))} \text{ reduce\_empty}$$

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = \Phi_{ub}^{\mathcal{P}}(p) \quad drift_\sigma(p) > 0}{(m, x, c, l) \xRightarrow{\text{reduce full } p} (m, x, c, reduce_p^{full}(l))} \text{ reduce\_full}$$

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = 0 \quad drift_\sigma(p) > 0 \quad restr_\sigma(p)}{(m, x, c, l) \xRightarrow{\text{reset empty } p} (m, x, c, reset_p(l))} \text{ reset\_empty}$$

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = \Phi_{ub}^{\mathcal{P}}(p) \quad drift_\sigma(p) < 0 \quad restr_\sigma(p)}{(m, x, c, l) \xRightarrow{\text{reset full } p} (m, x, c, reset_p(l))} \text{ reset\_full}$$

The premises of these rules are in direct agreement with the conditions of the four if-branches of Algorithm 1, as are the effects of the rules compared to the body of the if-branches. Therefore, the application of one rule corresponds to one execution of the loop body. As soon as a continuous transition is not fireable anymore, we reset the corresponding entries in the restriction lists:

$$\frac{t \in \mathcal{T}^{cont} \quad \neg fireable_\sigma(t) \quad p \in \mathcal{P}^{cont} \quad l(t)(p) < 1}{(m, x, c, l) \xRightarrow{\text{reset } t, p} (m, x, c, l[t, p \mapsto 1])} \text{ reset\_single}$$

Conflicts among these rules are again resolved probabilistically. Extending the method described in Section 3.2.1, we assume the rate adaption rules to have a weight of one, while the rules for firing of deterministic transitions are assigned the weight of the respective transition. Any possible scheduler can be motivated depending on the concrete system requirements. For example, we could always prefer the firing of transitions over rate adaption steps. We choose to resolve conflicts probabilistically.

Passing of Time. Lastly, the state of $\mathcal{H}$ can change as time passes. This has the effect that fireable continuous transitions fire and clocks of discrete transitions with concession evolve. However, we only allow time to pass as long as

1. no discrete transition is fireable (which implies that clocks do not run out),

2. the enabling status of continuous transitions does not change,
3. no continuous place over- or underflows, and
4. rate adaption is not necessary.

More concretely, time can only pass as long as no other semantic rule can be applied. We formalize this in the notion of a *safe time span*. For this, we first define the concrete sets of transitions affected by the passing of τ time units.

DEFINITION 3.16. **Fireable Continuous Transitions in Time Span**

Let $\mathcal{H}$ be a HPnG, $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $\tau \in \mathbb{R}_{>0}$. We write $\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau} \subseteq \mathcal{T}^{\mathit{cont}}$ for the set of fireable continuous transitions in the time span of duration τ starting from σ. Formally, we define $\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau} \subseteq \mathcal{T}^{\mathit{cont}}$ to be the largest set T such that for all $t \in T$ and $\sigma' = (m, \mathit{fire}^{\tau'}_{T}(x), c, l)$ with $\tau' \in (0, \tau)$, we have $\mathit{fireable}_{\sigma'}(t)$.

The set $\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}$ consists of the transitions that are fireable in the complete time span under consideration, and therefore those that are firing. Note that this is equal to the set of continuous transitions with concession in the time span, as a continuous transition is fireable if and only if it has concession. We can compute the set $\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}$ by starting with all continuous transitions and then removing those for which there exists a τ' where the transition is not fireable in the intermediate state σ'. We disregard the evolution of clocks in this definition, even though they would also change in the time span. This can be neglected in the above definition since the clock values do not influence whether a continuous transition is enabled or not. enabling status of continuous transitions does not depend on the clock values. Nevertheless, we need to include this later in order to describe the overall evolution of states in a time span, so we now define the set of discrete transition with concession in the time interval.

DEFINITION 3.17. **Discrete Transitions with Concession in Time Span**

Let $\mathcal{H}$ be a HPnG, $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $\tau \in \mathbb{R}_{>0}$. We write $\mathit{conc}^{\mathit{disc}}_{\sigma,\tau}$ for the set of discrete transitions with concession in the time span of duration τ starting in σ. starting from σ. Formally, we have $t \in \mathit{conc}^{\mathit{disc}}_{\sigma,\tau} \subseteq \mathcal{T}^{\mathit{cont}}$ iff for all $\sigma' = (m, \mathit{fire}^{\tau'}_{\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}}(x), c, l)$ with $\tau' \in (0, \tau)$, we have $\mathit{conc}_{\sigma'}(t)$.

The intermediate states σ' are defined in the same way as in Definition 3.16. Again, we can disregard the evolution of the clocks as the concession of discrete transition is also independent of the clocks. The following definition summarizes the conditions we require to hold in order to let τ time units pass from a state. As was described at the beginning of this paragraph, this includes making sure that no other semantic rule is applicable. Additionally, we add a condition to ensure the well-definedness of the sets $\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}$ and $\mathit{conc}^{\mathit{disc}}_{\sigma,\tau}$.

DEFINITION 3.18. **Safe Time Span**

Let $\mathcal{H}$ be a HPnG and $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$.

Then, we say that $\tau \in \mathbb{R}_{>0}$ is a *safe time span* from σ in $\mathcal{H}$ if for all $\tau' \in (0, \tau)$ and corresponding intermediate states $\sigma' = (m, x', c', l) = (m, \mathit{fire}^{\tau'}_{\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}}(x), \mathit{evolve}^{\tau'}_{\mathit{conc}^{\mathit{disc}}_{\sigma,\tau}}(c), l)$ all of the following conditions hold:

1. $\neg \mathit{fireable}_{\sigma'}(t)$ for all $t \in \mathcal{T}^{\mathit{disc}}$,
2. $x'(p) = 0 \implies \mathit{drift}_{\sigma'}(p) = 0 \vee (\mathit{drift}_{\sigma'}(p) > 0 \wedge \neg \mathit{restr}_{\sigma'}(p))$ for all $p \in \mathcal{P}^{\mathit{cont}}$,
3. $x'(p) = \Phi^{\mathcal{P}}_{\mathit{ub}}(p) \implies \mathit{drift}_{\sigma'}(p) = 0 \vee (\mathit{drift}_{\sigma'}(p) < 0 \wedge \neg \mathit{restr}_{\sigma'}(p))$ for all $p \in \mathcal{P}^{\mathit{cont}}$,
4. $\neg \mathit{fireable}_{\sigma'}(t) \implies l(t)(p) = 1$ for all $t \in \mathcal{T}^{\mathit{cont}}, p \in \mathcal{P}^{\mathit{cont}}$, and
5. for all $\tau'' \in (0, \tau)$ and corresponding intermediate states $\sigma'' = (m, x'', c'', l) = (m, \mathit{fire}^{\tau''}_{\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}}(x), \mathit{evolve}^{\tau''}_{\mathit{conc}^{\mathit{disc}}_{\sigma,\tau}}(c), l)$ we have
   a) $\mathit{fireable}_{\sigma''}(t) = \mathit{fireable}_{\sigma'}(t)$ for all $t \in \mathcal{T}^{\mathit{cont}}$ and
   b) $\mathit{conc}_{\sigma''}(t) = \mathit{conc}_{\sigma'}(t)$ for all $t \in \mathcal{T}^{\mathit{disc}}$.

We define $\mathit{safe}_{\sigma}(\tau)$ to hold if τ is a safe time span from σ.

The first four conditions ensure that none of the other semantic rules become applicable within τ time units from σ. They are equivalent to the negation of the premises of every semantic rule presented above. More precisely, the first condition states that no discrete transition becomes fireable. The second one expresses that if a place p is empty, it must either have a drift of zero, or a drift in the right direction and set no restrictions. This makes sure that no rate adaption step is necessary, because we have to update the rates as soon as the drift of p goes in the wrong direction. We have to reset the rates if the drift of p goes in the right direction, but p still places restrictions. Condition three is the dual requirement for places at the upper boundary. The fourth condition guarantees that the list of restrictions is maintained correctly, i.e., that the application of the *restriction* rule is not required.

Finally, the fifth condition guarantees that the sets $\mathit{fireable}^{\mathit{cont}}_{\sigma,\tau}$ and $\mathit{conc}^{\mathit{disc}}_{\sigma,\tau}$ are stable in the sense that they do not change within $(0, \tau)$. This is done by making sure that neither the enabling status of continuous transitions nor the concession status of discrete transitions changes, but remains the same for all intermediate states σ' and σ''.

The time interval $(0, \tau)$ we are considering is open instead of closed. This is because, for example, if a discrete transition has concession until time 0 and again from time τ on, it still makes sense to consider τ to be safe. We also want to allow discrete transitions to become fireable at the end of the interval. Therefore, we only require the properties

to hold for intermediate states within $(0, \tau)$.

We can infer some properties of the intermediate states in a safe time span. In particular, Theorem 3.2 states that the levels of all continuous places are guaranteed to remain within their boundaries.

THEOREM 3.2. **Properties of a Safe Time Span**

Let $\mathcal{H}$ be a HPnG, $\sigma \in \Sigma^{\mathcal{H}}$ a state of $\mathcal{H}$, and $\tau \in \mathbb{R}_{>0}$ a safe time span. For $\tau' \in (0, \tau)$ and the corresponding intermediate state $\sigma' = (m, x', c', l)$ where $x' = fire^{\tau'}_{fireable^{cont}_{\sigma',\tau'}}(x)$ and $c' = evolve^{\tau'}_{conc^{disc}_{\sigma',\tau'}}(c)$, it holds that $0 \leq x'(p) \leq \Phi^{\mathcal{P}}_{ub}(p)$ for all $p \in \mathcal{P}^{cont}$.

Proof. Since τ is a safe time span, we know that $x'(p) = 0 \implies drift_{\sigma'}(p) \geq 0$ and dually that $x'(p) = \Phi^{\mathcal{P}}_{ub}(p) \implies drift_{\sigma'}(p) \leq 0$. We also know that $fire^{\tau'}_{fireable^{cont}_{\sigma',\tau'}}(x)$ updates the fluid levels of continuous places according to the current drift. Since the drift is always in the right direction, we know that if $x'(p) = 0$, the level of place p can only rise and dually if $x'(p) = \Phi^{\mathcal{P}}_{ub}(p)$, the level can only fall. □

Now, we can give the rule for a time step.

$$\frac{\tau \in \mathbb{R}_{>0} \quad safe_{\sigma}(\tau)}{(m, x, c, l) \xrightarrow{\tau} (m, fire^{\tau}_{fireable^{cont}_{\sigma,\tau}}(x), evolve^{\tau}_{conc^{disc}_{\sigma,\tau}}(c), l)} \text{ time}$$

If τ is a safe time span from σ, the continuous markings evolve as specified by by Definition 3.9 and the clocks corresponding to discrete transitions are decreased as specified by Definition 3.10. As the premise of this rule explicitly ensures the other rules to be non-applicable, there is no need for resolving conflicts including this rule.

Paths of HPnGs. The seven inference rules presented above give a complete characterization of the behavior of HPnGs. As for SHAs, we now define execution steps and paths of HPnGs.

DEFINITION 3.19. **Operational Semantics of HPnGs**

Let $\mathcal{H}$ be a HPnG. The semantics of $\mathcal{H}$ is given by the rules for firing deterministic transitions, rate adaption, and the passing of time. We can take an *execution step*,

denoted by $\Rightarrow$, by applying one of these rules, i.e., we define

$$\Rightarrow \quad = \quad \bigcup_{t \in \mathcal{T}^{disc}} \xRightarrow{\text{fire } t}$$
$$\cup \bigcup_{p \in \mathcal{P}^{cont}} \xRightarrow{\text{reduce full } p} \cup \xRightarrow{\text{reduce empty } p} \cup \xRightarrow{\text{reset full } p} \cup \xRightarrow{\text{reset full } p}$$
$$\cup \bigcup_{t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}} \xRightarrow{\text{reset } t, p}$$
$$\cup \bigcup_{\tau \in \mathbb{R}_{>0}} \xRightarrow{\tau}$$

A *path* of $\mathcal{H}$ is an infinite sequence $\sigma_0, \sigma_1, \sigma_2, \ldots$ of states of $\mathcal{H}$ such that

$$\sigma_0 \Rightarrow \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots$$

and σ_0 is an initial state. Under abuse of notation, we also write $\sigma \Rightarrow \sigma'$ if σ' is reachable from σ taking several execution steps.

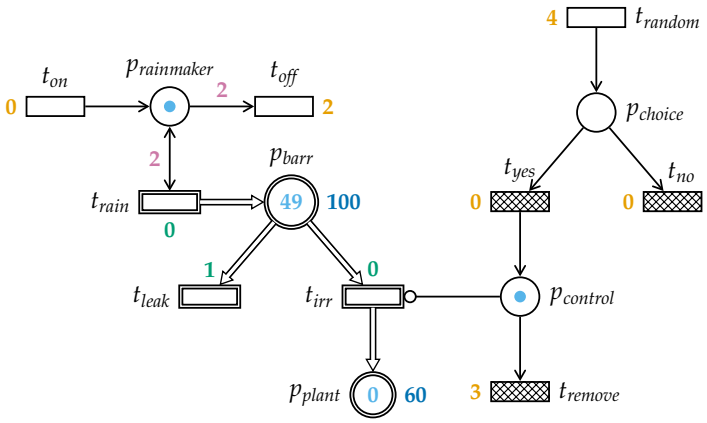
EXAMPLE 3.8. **Operational Semantics of HPnGs**

We again consider the HPnG $\mathcal{H}$ from Example 3.1 depicted in Figure 3.2. Figure 3.12 shows the first four states of a path of $\mathcal{H}$, starting in the initial state discussed in Example 3.2 and depicted in Figure 3.4. The prefix of the path is formally given by

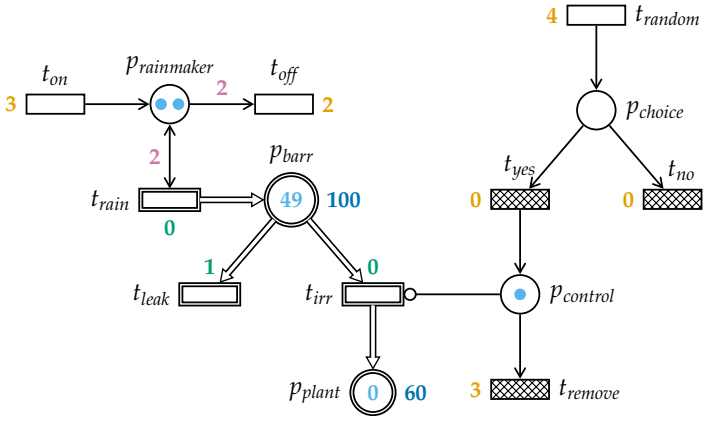
$$\sigma_0 \xRightarrow{\tau = 1} \sigma_1 \xRightarrow{\text{fire } t_{on}} \sigma_2 \xRightarrow{\tau = 2} \sigma_3.$$

with the states σ_i as specified in Table 3.1. In the first step, one time unit passes. The level of the barrel is decreased by one as the transition modeling the leak is fireable and fires with a rate of one liter per hour. Also, the clocks of the transitions with concession are decreased by one. Note that t_{off} does not have concession because the weight of the connecting arc is two, so it requires two tokens in $p_{rainmaker}$ to have concession. Therefore, in σ_1, the clock of t_{on} is zero. The transition fires and moves another token to $p_{rainmaker}$. Additionally, the clock value for t_{on} is resampled, set to four and we end up in σ_2.

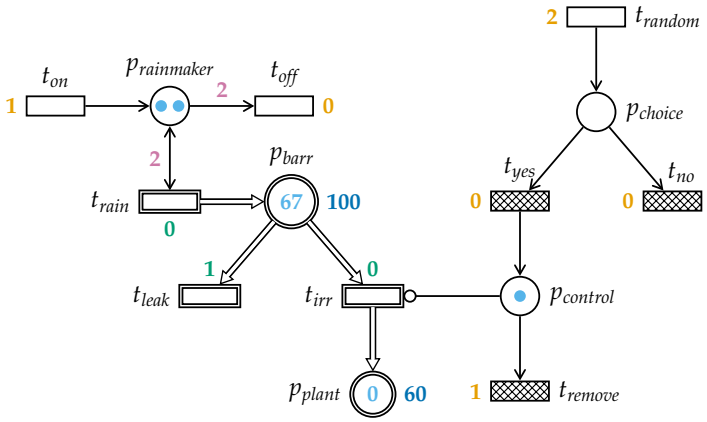
Then, we let two time units pass. Since the transition t_{rain} is now fireable as there are two tokens in $p_{rainmaker}$, twenty liters are added to the barrel, while the leak removes two, so the level of p_{barr} in σ_3 is 67. Notice that now, the clock of t_{off} is zero, so the transition fires, and we could not have taken a longer time step.



(a) State θ_1 reached from θ_0 by letting one time unit pass.



(b) State θ_2 reached from θ_1 by the firing of transition t_{on}.



(c) State θ_3 reached from θ_2 by letting two time units pass.

Figure 3.12.: Path of the HPnG $\mathcal{H}$ as discussed in Example 3.8.

State	Discrete Markings m	Fluid Markings x	Clocks c	Restrictions l
σ_0	$p_{rainmaker} \mapsto 1$ $p_{control} \mapsto 1$ $p_{choice} \mapsto 0$	$p_{barr} \mapsto 50$ $p_{plant} \mapsto 0$	$t_{remove} \mapsto 4$ $t_{yes} \mapsto 0$ $t_{no} \mapsto 0$ $t_{on} \mapsto 1$ $t_{off} \mapsto 2$ $t_{random} \mapsto 5$	**1**
σ_1	$p_{rainmaker} \mapsto 1$ $p_{control} \mapsto 1$ $p_{choice} \mapsto 0$	$p_{barr} \mapsto \mathbf{49}$ $p_{plant} \mapsto 0$	$t_{remove} \mapsto \mathbf{3}$ $t_{yes} \mapsto 0$ $t_{no} \mapsto 0$ $t_{on} \mapsto \mathbf{0}$ $t_{off} \mapsto 2$ $t_{random} \mapsto \mathbf{4}$	**1**
σ_2	$p_{rainmaker} \mapsto \mathbf{2}$ $p_{control} \mapsto 1$ $p_{choice} \mapsto 0$	$p_{barr} \mapsto 49$ $p_{plant} \mapsto 0$	$t_{remove} \mapsto 3$ $t_{yes} \mapsto 0$ $t_{no} \mapsto 0$ $t_{on} \mapsto \mathbf{3}$ $t_{off} \mapsto 2$ $t_{random} \mapsto 4$	**1**
σ_3	$p_{rainmaker} \mapsto 2$ $p_{control} \mapsto 1$ $p_{choice} \mapsto 0$	$p_{barr} \mapsto \mathbf{67}$ $p_{plant} \mapsto 0$	$t_{remove} \mapsto \mathbf{1}$ $t_{yes} \mapsto 0$ $t_{no} \mapsto 0$ $t_{on} \mapsto \mathbf{1}$ $t_{off} \mapsto \mathbf{0}$ $t_{random} \mapsto \mathbf{2}$	**1**

Table 3.1.: The prefix of a path of $\mathcal{H}$ introduced in Example 3.1. Updated values are highlighted. The execution steps leading from σ_i to σ_{i+1} are discussed in Example 3.8.

4. Transforming HPnGs to SHAs

In the previous chapter, we saw that Petri nets are very illustrative: Their components can be presented in a simple and clear manner and the structure of the modeled system is easily recognizable. However, the analyzing techniques for hybrid automata are more evolved. For example, the analysis of reachability probabilities for a specific set of states is an active area of research that has produced several advanced tools [Alt15; CÁS12; Fre+11]. To profit from these results, one could either adapt the proposed techniques to HPnGs [PR17; HR19; Hül+21], or define a transformation from HPnGs to SHAs [Pil+20]. The latter has the strong advantage that all available methods are applicable without further ado, whereas the former requires an individual treatment of each method. In this chapter, we define a transformation of HPnGs to SHAs, thus profiting from both the visual simplicity of HPnGs and the advanced analyzing techniques of SHAs.

Related Work. A precursor of transforming HPnGs to SHAs is given by transforming their non-stochastic variants. An early algorithmic transformation of hybrid Petri nets to hybrid automata was given by Allam and Alla, though lacking a proof of correctness [AA98; DA01]. Based on this, a structural transformation was defined and proven to be correct by Ghomri and Alla [GA18]. Further, there exists a transformation of generalized stochastic Petri nets to Markov automata [Eis+13], which are essentially non-deterministic continuous-time Markov chains.

A transformation of HPnGs to a subclass of SHAs was presented and implemented by Pilch et al. [Pil+20]. Their approach uses a symbolic representation of paths of the HPnG up to a given time bound and therefore produces an SHA that models bounded executions of the HPnG. We aim to avoid symbolic computations and instead give a direct transformation of HPnGs to SHAs that can simulate infinite paths.

Possible Approaches and Considerations. A fundamental difference between SHAs and HPnGs is in their underlying assumptions made about time. If an invariant is violated and no jump is possible, time can stop in an SHA. This is not possible in HPnGs: There are no invariants, and time can always pass. Arising conflicts that would violate given specifications, such as the upper bound of a continuous place, are then solved, for example, by rate adaption. In the transformed SHA, we therefore need to ensure that if an invariant is violated, it is possible to move to another location to guarantee that

the behavior of the HPnG is accurately modeled. Additionally, we also want to avoid unrealistic behavior as discussed in Section 2.2. This correlates with the point made before, as we have to ensure that timelocks and Zeno behavior are excluded.

When trying to define a transformation of HPnGs, the first question that arises is how one can model the places of a HPnG. Since there are infinitely many possible configurations for the number of tokens and the amount of fluid in the places, modeling these assignments with locations would require infinitely many locations. Pilch et al. model the number of discrete tokens using a finite number of locations and consequently only consider bounded executions of the HPnG [Pil+20]. We want to model the entire behavior of a HPnG, suggesting that discrete and continuous places should be modeled using variables specifying the number of tokens and the amount of fluid.

With the markings stored in variables, this raises the question of how we can model the firing of transitions. For discrete transitions, the answer is quite straightforward. We can define a jump that manipulates the marking in the same way as the transition, guarded by conditions that make sure that the jump is enabled if and only if the transition has concession. Then, the corresponding clock is assigned a suitable probability distribution, making the jump fireable exactly when the transition is fireable. The firing of continuous transitions, however, cannot be modeled using discrete jumps because they continuously move fluid between places. Thus, we need the corresponding variables to evolve continuously. To realize this, we make use of the continuous component of SHAs and define the derivatives of a location in such a way that it matches the evolution of each place's level in the HPnG.

Conversely, one must also consider what the locations of the SHA should represent. In general, locations can be used to encode different evolutions of variables by assigning different activities per location. Therefore, it would be conceivable to let a location reflect whether a continuous transition is enabled or not, since this has an effect on the evolution of the variables modeling its neighboring places. However, we can also simply describe the evolution of the fluid held by a place via its drift, which can be calculated from the firing rates of neighboring transitions.

Another possibility would be to use locations to translate the rate adaption algorithm. In this scenario, each continuous place p is modeled using three locations: One location is representing that the place is within its boundaries and we are not in the process of rate adaption, meaning that the place currently does not restrict any transitions. Another location represents p being empty and restricting transitions. The third location dually represents p being full and restricting transitions. Even though this approach lets us derive the current status of a place in the rate adaption algorithm, there are no decisive advantages over modeling everything in one location. On the contrary, this approach requires several additional jumps that can otherwise be avoided. We ultimately came to the conclusion that there is no need for multiple locations to properly model the

behavior of the HPnG.

One could argue that a disadvantage of modeling everything in one location is the loss of the graphical structure of the HPnG, making the automaton harder to understand, as will be evident in the presented examples. Nevertheless, we decided to go with this approach since the purpose of this transformation is the (automatic) analysis, and human readability is not the main focus.

As was already mentioned several times, we want to define the transformation in a compositional way. For this, we have to decide how to divide the HPnG into smaller parts in a reasonable way. Each part will then be transformed separately and the resulting sub-SHAs will be composed. One approach is to translate each place and transition separately. This would give us a large number of sub-SHAs, which makes the composition more complex than necessary. Instead, it makes sense to bundle some parts. The other extreme would be to translate everything at once without requiring any composition at all. In this case, the resulting definition would be hard to automatize, very extensive, affecting readability and making it difficult to understand. Consequently, it also makes sense to split some parts.

Therefore, we now discuss which parts we should consider separately. At a fairly early stage, we came to the conclusion that each discrete fragment can be bundled up and translated at once. This is mainly because we can model their behavior by adding one jump per transition, which is a manageable number. For the continuous part, we have to include the rate adaption algorithm. This requires several jumps per continuous place, as we will see later, and is therefore more advanced than the discrete part. As a consequence, we decided not to transform the continuous part at once, but split it further. Now, we could either consider each place or each transition separately. Recall that we want to model the level of each place in a variable. For the second option, this variable would have to be controlled by several SHAs corresponding to continuous transitions. However, this can be avoided when defining one SHA per place and we concluded that it is best to translate each continuous place separately.

Our Approach. In the transformation defined in this thesis, each discrete fragment is translated into one location. The firing of transitions is modeled by taking a jump that modifies the markings accordingly. The intrinsic handling of clocks in SHAs will be used to model the clocks in HPnGs. Additionally, each continuous place is transformed into one separate location. All jumps are selfloops and model the rate adaption and maintain the list of restrictions. For urgent jumps, we will use Dirac distributions δ_a for $a \in \mathbb{R}_{\geq 0}$, making sure that the jump is taken as soon as it becomes enabled. Finally, we compose all sub-SHAs to obtain an SHA that is semantically equivalent to the HPnG. Since all sub-SHAs are only made of one location, so is the composed SHA, and there is no blow-up.

Note that we do not need label synchronization in this setting. Instead, the sub-SHAs communicate through shared variables that encode the restrictions set on the transitions by the places, which influence the actual firing rate of the transitions and thereby the drift of each place.

Throughout the chapter, we will translate several of the defined predicates in Chapter 3 to the transformation framework. We start with defining the set of variables in Section 4.1. In Section 4.2, we define the transformation of discrete fragments and continue with the transformation of continuous places in Section 4.3. We combine these two into a compositional transformation of HPnGs to SHAs in Section 4.4. A proof of correctness via bisimulation is provided in Section 4.5.

4.1. Preliminary Definitions

For the remainder of this chapter, let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG. From $\mathcal{H}$, we want to obtain an SHA $\mathcal{A}^{\mathcal{H}}$ simulating the behavior of $\mathcal{H}$. As elaborated above, we will split $\mathcal{H}$ into smaller parts and transform them separately. Each sub-SHA will be defined over the same set of variables defined in what follows.

Definition 4.1. **Variable and Valuation Set for HPnGs**
Given a HPnG $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$, we define $Var^{\mathcal{H}} = M \cup X \cup L$ with

- $M = \{m_p \mid p \in \mathcal{P}^{disc}\}$,
- $X = \{x_p \mid p \in \mathcal{P}^{cont}\}$, and
- $L = \{l_{t,p} \mid p \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont}\}$.

We denote by $\mathcal{V}^{\mathcal{H}}$ the set of valuations over $Var^{\mathcal{H}}$.

For each place p of a HPnG $\mathcal{H}$, we define a variable m_p or x_p, depending on the type of the place, that represents the current marking from a state (m, x, c, l) of $\mathcal{H}$. Furthermore, we save the list of restrictions using variables, i.e., the entry $l(t)(p)$ for a transition t and a place p corresponds to the variable $l_{t,p}$. Note that the clocks, while being a part of the state of $\mathcal{H}$ (see Definition 3.4), do not have to be explicitly modeled here. Instead, we make use of the clocks assigned to the labels of an SHA.

We now translate some predicates that were introduced in Section 3.2. The idea is to express the same properties using a valuation of the variable set $\mathcal{V}^{\mathcal{H}}$. We start by defining when a transition has concession with respect to such a valuation.

Definition 4.2. Concession

Let $\mathcal{H}$ be a HPnG and $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation of the corresponding variables. A transition $t \in \mathcal{T}$ has *concession* in ν iff

1. the discrete places connected to t via a discrete arc contain enough tokens, i.e.,
$$\forall p \in \mathcal{I}^{disc}(t) \colon \nu(m_p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle,$$
2. the discrete places connected via an inhibitor arc do not contain more tokens than allowed, i.e.,
$$\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{disc}) \colon \nu(m_p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle,$$
3. the continuous places connected via an inhibitor arc do not contain more fluid than allowed, i.e.,
$$\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{cont}) \colon \nu(x_p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle,$$
4. the discrete places connected via an test arc do not violate the testing condition, i.e.,
$$\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{disc}) \colon \nu(m_p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle, \text{ and}$$
5. the continuous places connected via an test arc do not violate the testing condition, i.e.,
$$\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{cont}) \colon \nu(x_p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle.$$

We define $conc_\nu(t)$ to hold iff t has concession in ν.

This exactly corresponds to Definition 3.6, except that we compare $\nu(m_p)$ and $\nu(x_p)$ instead of $m(p)$ and $x(p)$ to the parameters. We will use the predicate $conc_\nu(t)$ as guard for the jump modeling the firing of a discrete transition t. The effect of the jump, i.e., the movement of tokens, should be analogous to what was defined for HPnGs in Definition 3.9. In contrast to this, the firing of continuous transitions is handled by activities assigned to locations. Thus, we define the following predicate defining the effect of firing only for discrete transitions.

Definition 4.3. Transformed Valuation By Firing of Discrete Transitions

Let $\mathcal{H}$ be a HPnG, $t \in \mathcal{T}^{disc}$, and $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation such that t has concession in ν, i.e., $conc_\nu(t)$ holds. We define the transformed valuation by firing of t as

$fire_\nu(t) \in \mathcal{V}^{\mathcal{H}}$, where

$$fire_\nu(t)(y) = \begin{cases} \nu(m_p) - \Phi_w^{\mathcal{A}}\langle p, t\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{I}^{disc}(t) \\ \nu(m_p) + \Phi_w^{\mathcal{A}}\langle t, p\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{O}^{disc}(t) \\ \nu(y) & \text{else} \end{cases}$$

for all $y \in Var^{\mathcal{H}}$.

Again, this exactly corresponds to Definition 3.9 with the difference that the markings are taken from a valuation ν over $Var^{\mathcal{H}}$ instead of a state σ of the HPnG.

4.2. Transforming Discrete Fragments

As defined in Definition 3.2, a discrete fragment $\mathcal{D}$ is made of

- a set of discrete places $\mathcal{D} \subseteq \mathcal{P}^{disc}$,
- the set of discrete transitions $\mathcal{T}^{\mathcal{D}}$ connected to these places, and
- the set of arcs $\mathcal{A}^{\mathcal{D}}$ that are either discrete and within the fragment, or inhibitor and test arcs both within the fragment and connecting transitions from the fragment to continuous places.

We use a single location *loc* to model one discrete fragment. All places in $\mathcal{D}$ are modeled by variables M that store the current number of tokens for each place, as defined in Definition 4.1. The firing of a transition t is then modeled as a jump $jump_t$ from *loc* to *loc* whose guard ensures that the transition has concession, and the effect describes how exactly the tokens are moved.

In the HPnG $\mathcal{H}$, the firing time of a discrete transition $t \in \mathcal{T}^{disc}$ is either fixed by the parameter $\Phi_{ft}^{\mathcal{T}}(t)$ if t is deterministic or sampled from the probability distribution $\Phi_{gt}^{\mathcal{T}}(t)$ if t is general. The firing time of a deterministic transition can also be expressed as a probability distribution, more precisely as the Dirac distribution $\delta_{\Phi_{ft}^{\mathcal{T}}(t)}$. Since $jump_t$ models the firing of t, we assign the distributions to $jump_t$ via labels. Consequently, the firing time of t is then equal to the enabling duration of $jump_t$. In the following formal definition, we omit the jump identifiers and implicitly assign each jump a unique value.

The semantics of the arcs is already reflected in the above considerations and does not have to be translated separately. More precisely, discrete arcs specify how many tokens are moved by the transitions and therefore are contained in the effect of each $jump_t$. The inhibitor and test arcs add conditions to the concession status of transitions, which are part of the guard of the jumps.

Definition 4.4. Transforming Discrete Fragments

Let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG and $\mathcal{D}$ be a discrete fragment of $\mathcal{H}$ with transitions $\mathcal{T}^{\mathcal{D}}$ and arcs $\mathcal{A}^{\mathcal{D}}$. Then, we define a corresponding SHA $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}}$ as

$$\mathcal{A}^{\mathcal{H}}_{\mathcal{D}} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

with the following components:

- $Loc = \{loc\}$.
- $Var = Var^{\mathcal{H}}$ with $Con = \{m_p \in M \mid p \in \mathcal{D}\}$ and $NCon = Var \backslash Con$.
- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- $Init(loc) = \{\nu \in \mathcal{V}^{\mathcal{H}} \mid \forall m_p \in Con.\ \nu(m_p) = m_0(p)\}$.
- $Edge = \{jump_t \mid t \in \mathcal{T}^{\mathcal{D}}\}$ where $jump_t = (loc, \mu, loc)$ with

$$\mu = \left\{(\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; conc_\nu(t) \wedge \nu' = fire_\nu(t)\right\}.$$

- $Act(loc)$ is the set of activities that are solutions of $\dot{m}_p = 0$ for all $m_p \in Con$.
- $Lab = \{a_t \mid t \in \mathcal{T}^{\mathcal{D}}\}$.
- $Proc(jump_t) = a_t$ for all $t \in \mathcal{T}^{\mathcal{D}}$.
- For all $\nu \in \mathcal{V}^{\mathcal{H}}$,
  - $Dur(a_t, (loc, \nu)) = \Phi^{\mathcal{T}}_{gt}(t)$ for $t \in \mathcal{T}^{gen}$ and
  - $Dur(a_t, (loc, \nu)) = \delta_{\Phi^{\mathcal{T}}_{ft}(t)}$ for $t \in \mathcal{T}^{det}$.
- $Wgt(jump_t) = \Phi^{\mathcal{T}}_{p}(t)$.

As already indicated above, the SHA $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}}$ is made of one location loc. The SHA for a discrete fragment $\mathcal{D}$ controls exactly the variables corresponding to the markings of places in $\mathcal{D}$. All other variables from $Var^{\mathcal{H}}$ are non-controlled. The only condition that has to hold for the marking of a discrete place is that it is a natural number, which is implicitly ensured by the definition of the initial valuation and the jumps. Thus, we do not impose any invariant on the controlled variables. In the syntax of HPnGs, the marking m_0 gives the initial tokens contained in each place. Accordingly, the set of initial valuations contains the valuations where m_p is equal to the initial marking $m_0(p)$ of place p. For each discrete transition $t \in \mathcal{T}^{\mathcal{D}}$, the jump $jump_t$ models the firing of t. The jump is enabled if t has concession. In accordance with the notion of transitions being fireable from the semantics of HPnGs (Definition 3.7), $jump_t$ is fireable if t has concession and the corresponding clock is zero. We will at a later point give a formal

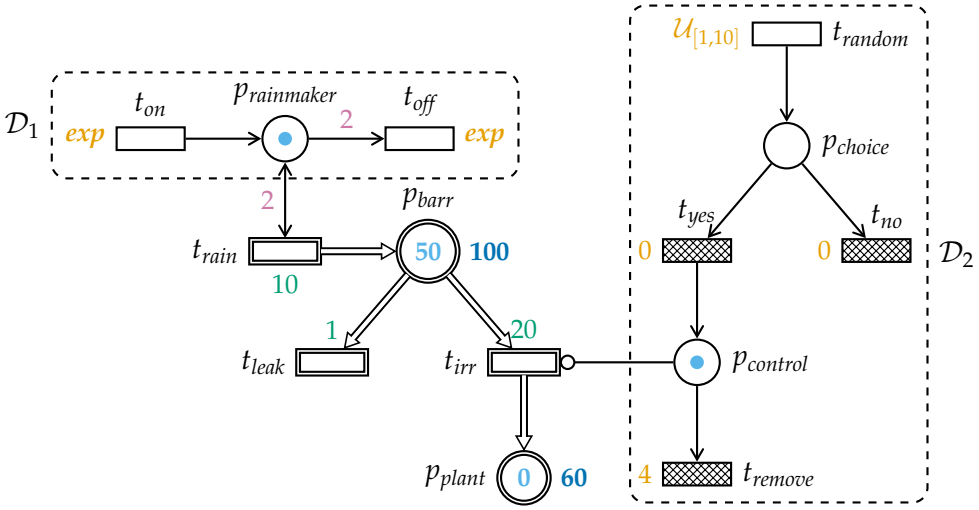


Figure 4.1.: The HPnG $\mathcal{H}$ modeling an irrigation system as introduced in Example 3.1 and further discussed in the course of Chapter 3.

proof of this equivalence. After taking $jump_t$, the new valuation is given by $fire_\nu(t)$ as defined in Definition 4.3. Since the markings of the discrete places only evolve by the firing of deterministic transitions, the activities for the controlled variables do not modify the controlled variables. Consequently, their derivative is set to zero.

For the stochastic part of the SHA, we assign a unique label to each jump (and thus indirectly to each transition). Then, the duration of a label is set to either $\Phi^{\mathcal{T}}_{gt}(t)$ if the corresponding transition is general or $\delta_{\Phi^{\mathcal{T}}_{ft}(t)}$ if the corresponding transition is deterministic. The weights of the jumps are set to the priority $\Phi^{\mathcal{T}}_p(t)$ of the corresponding transition. As a consequence, a conflict between jumps in $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}}$ is resolved as the conflict between the corresponding transitions in the HPnG (Section 3.2.1). More precisely, the probability of taking $jump_t$ in a state $\sigma \in \Sigma^{\mathcal{H}}$ of $\mathcal{H}$ is given by

$$\frac{Wgt(jump_t)}{\sum_{jump_{t'} \in Edge\_fireable_\vartheta(jump_{t'})} Wgt(jump_{t'})} = \frac{\Phi^{\mathcal{T}}_p(t)}{\sum_{t' \in \mathcal{T}^{disc}\_fireable_\sigma(t')} \Phi^{\mathcal{T}}_p(t')}$$

where $\vartheta \in \Theta^{\mathcal{A}^{\mathcal{H}}}$ is a state of the result of the transformation corresponding to σ. The exact translation of states will be given in Section 4.5. This equality holds as $Wgt(jump_t) = \Phi^{\mathcal{T}}_p(t)$ by definition and the notion of being fireable coincides for $\mathcal{H}$ and the result of the transformation, which will be proven later.

Throughout this chapter, we are going to gradually transform the HPnG $\mathcal{H}$ modeling an irrigation system discussed in Chapter 3, depicted again in Figure 4.1. We start with transforming both discrete fragments.

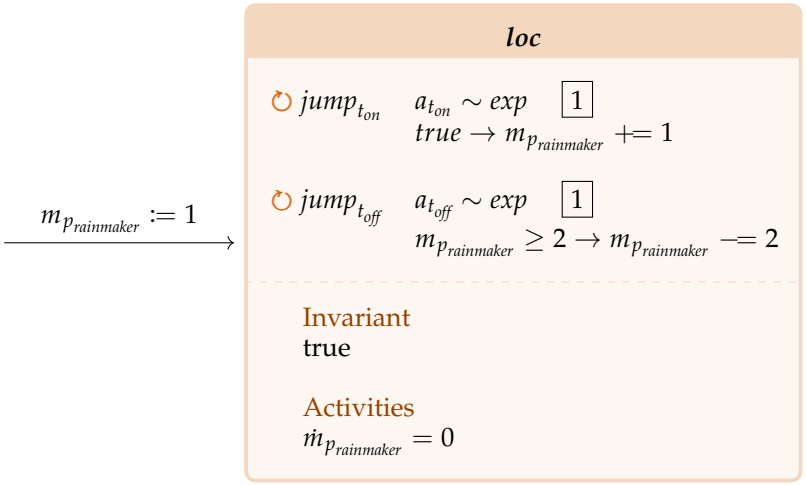


Figure 4.2.: Depiction of the SHA $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}_1}$ discussed in Example 4.1. We only have jumps from *loc* to *loc*. For better readability, we therefore denote the probability distribution, the weight, as well as the guard and the effect of each jump within the location, symbolized by ↻.

EXAMPLE 4.1. **Transforming Discrete Fragments**

Figure 4.1 depicts the HPnG $\mathcal{H}$ known from Chapter 3. $\mathcal{H}$ has two discrete fragments $\mathcal{D}_1$ and $\mathcal{D}_2$. We start by transforming $\mathcal{D}_1$, which only contains the place $p_{rainmaker}$ and the transitions t_{on} and t_{off}. As defined in Definition 4.1, the set of variables is given by $Var^{\mathcal{H}} = M \cup X \cup L$ with

- $M = \{m_{p_{rainmaker}}, m_{p_{control}}, m_{p_{choice}}\}$,
- $X = \{x_{p_{barr}}\}$, and
- $L = \{l_{t_{rain},p_{barr}}, l_{t_{leak},p_{barr}}, l_{t_{irr},p_{barr}}, l_{t_{rain},p_{plant}}, l_{t_{leak},p_{plant}}, l_{t_{irr},p_{plant}}\}$.

Adhering to Definition 4.4, the SHA corresponding to $\mathcal{D}_1$ is given by

$$\mathcal{A}^{\mathcal{H}}_{\mathcal{D}_1} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

where

- $Loc = \{loc\}$.
- $Var = Var^{\mathcal{H}}$ with $Con = \{m_{p_{rainmaker}}\}$ and $NCon = Var^{\mathcal{H}} \backslash Con$.
- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- $Init(loc) = \{\nu \in \mathcal{V}^{\mathcal{H}} \mid \nu(m_{p_{rainmaker}}) = m_0(p_{rainmaker}) = 1\}$.

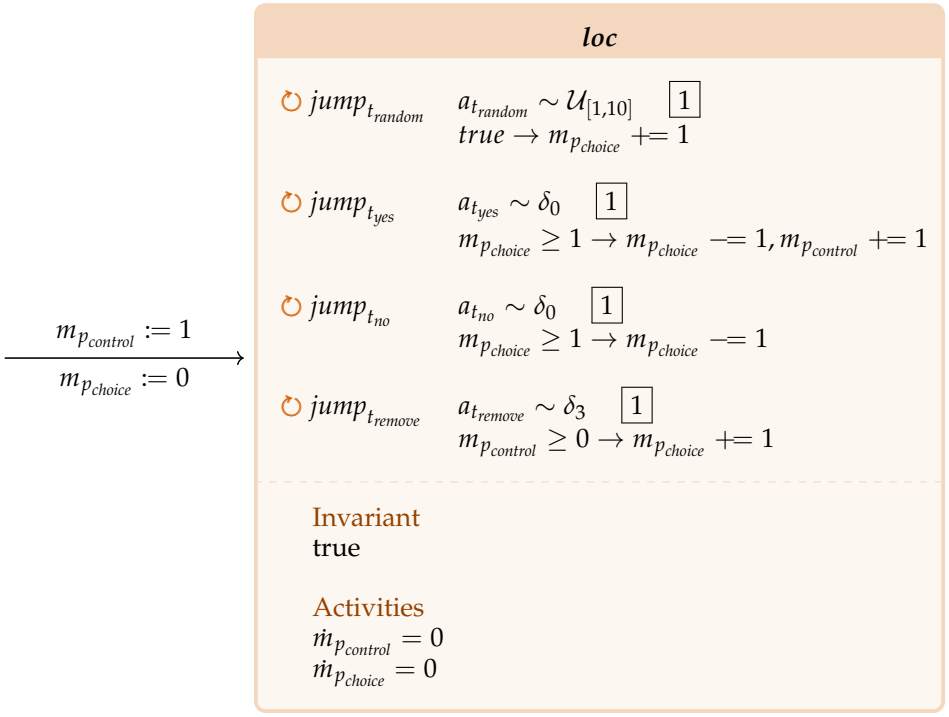


Figure 4.3.: Depiction of the SHA $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}_2}$ discussed in Example 4.1.

- $Edge = \left\{jump_{t_{on}}, jump_{t_{off}}\right\}$ where $jump_{t_{on}} = (loc, \mu_1, loc)$ with

$$\begin{aligned}\mu_1 &= \left\{(v, v') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; conc_v(t_{on}) \wedge v' = fire_v(t_{on})\right\} \\ &= \left\{(v, v') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; v' = v[m_{p_{rainmaker}} \mathrel{+}= 1]\right\}\end{aligned}$$

  and $jump_{t_{off}} = (loc, \mu_2, loc)$ with

$$\begin{aligned}\mu_2 &= \left\{(v, v') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; conc_v(t_{off}) \wedge v' = fire_v(t_{off})\right\} \\ &= \left\{(v, v') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; v(m_{p_{rainmaker}}) \geq 2 \wedge v' = v[m_{p_{rainmaker}} \mathrel{-}= 2]\right\}.\end{aligned}$$

- $Act(loc)$ is the set of activities that are solutions of

$$\dot{m}_{p_{rainmaker}} = 0.$$

- $Lab = \left\{a_{t_{on}}, a_{t_{off}}\right\}$.
- $Proc(jump_{t_{on}}) = a_{t_{on}}$ and $Proc(jump_{t_{off}}) = a_{t_{off}}$.

- $Dur(a_{t_{on}}, (loc, \nu)) = \Phi^{\mathcal{T}}_{gt}(a_{t_{on}}) = exp$ and $Dur(a_{t_{off}}, (loc, \nu)) = \Phi^{\mathcal{T}}_{gt}(a_{t_{off}}) = exp$ for $\nu \in \mathcal{V}^{\mathcal{H}}$.
- $Wgt(jump_{t_{on}}) = \Phi^{\mathcal{T}}_{p}(t_{on}) = 1$ and $Wgt(jump_{t_{off}}) = \Phi^{\mathcal{T}}_{p}(t_{off}) = 1$.

A visual representation of $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}_1}$ is given in Figure 4.2. As can be seen, the only location loc controls the variable $m_{p_{rainmaker}}$ modeling the marking of the place $p_{rainmaker}$. The initial valuation makes sure that we can only start in a state where the marking is one, as defined in the initial marking $m_0(p_{rainmaker}) = 1$.

For both transitions t_{on} and t_{off} from the discrete fragment $\mathcal{D}_1$ there is one jump representing the firing of the transition. We take a closer look at t_{off}. In the HPnG, the transition only has an incoming place, namely $p_{rainmaker}$. The connected arc $\langle p_{rainmaker}, t_{off} \rangle$ has a weight of one. Therefore, t_{off} has concession if there is at least one token in $p_{rainmaker}$. This is realized in the guard $\nu(m_{p_{rainmaker}}) \geq 1$. When the transition fires, it removes one token from $p_{rainmaker}$. This exactly is expressed in the effect of $jump_{t_{off}}$ by requiring that $\nu' = \nu[m_{p_{rainmaker}} \mathrel{-}= 1]$. The activities do not modify $m_{p_{rainmaker}}$.

The probabilistic components basically assign the exponential distribution exp to both jumps $jump_{t_{on}}$ and $jump_{t_{off}}$. As a consequence, the enabling duration of both transitions is exponentially distributed, corresponding to the firing times in $\mathcal{H}$. The weight of the jumps also matches the weight of the transitions.

The second discrete fragment $\mathcal{D}_2$ contains the places $p_{control}$, p_{choice} and the transitions t_{random}, t_{yes}, t_{no}, and t_{remove}. The SHA corresponding to $\mathcal{D}_2$ is given by

$$\mathcal{A}^{\mathcal{H}}_{\mathcal{D}_2} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

where

- $Loc = \{loc\}$.
- $Var = Var^{\mathcal{H}}$ with $Con = \{m_{p_{control}}, m_{p_{choice}}\}$ and $NCon = Var \backslash Con$.
- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- $Init(loc) = \left\{ \nu \in \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(m_{p_{control}}) = 1 \wedge \nu(m_{p_{choice}}) = 0 \right\}$.
- $Edge = \{jump_{t_{random}}, jump_{t_{yes}}, jump_{t_{no}}, jump_{t_{remove}}\}$ where $jump_{t_{random}} = (loc, \mu_1, loc)$ with

$$\mu_1 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu' = \nu[m_{p_{choice}} \mathrel{+}= 1] \right\},$$

$jump_{t_{yes}} = (loc, \mu_2, loc)$ with

$$\mu_2 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(m_{p_{choice}}) \geq 1 \wedge \nu' = \nu[m_{p_{choice}} \mathrel{-}= 1, m_{p_{control}} \mathrel{+}= 1] \right\},$$

$jump_{t_{no}} = (loc, \mu_3, loc)$ with

$$\mu_3 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(m_{p_{choice}}) \geq 1 \wedge \nu' = \nu[m_{p_{choice}} \mathrel{-}= 1] \right\}, \text{and}$$

$jump_{t_{remove}} = (loc, \mu_4, loc)$ with

$$\mu_4 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(m_{p_{control}}) \geq 1 \wedge \nu' = \nu[m_{p_{choice}} \mathrel{+}= 1] \right\}.$$

- $Act(loc)$ is the set of activities that are solutions of

$$\dot{m}_{p_{control}} = 0 \quad \text{and} \quad \dot{m}_{p_{choice}} = 0.$$

- $Lab = \{a_{t_{random}}, a_{t_{yes}}, a_{t_{no}}, a_{t_{remove}}\}$.
- $Proc$ is given by
  - $Proc(jump_{t_{random}}) = a_{t_{random}}$,
  - $Proc(jump_{t_{yes}}) = a_{t_{yes}}$,
  - $Proc(jump_{t_{no}}) = a_{t_{no}}$, and
  - $Proc(jump_{t_{remove}}) = a_{t_{remove}}$.
- Dur is given by
  - $Dur(a_{t_{random}}, (loc, \nu)) = \mathcal{U}_{[1,10]}$,
  - $Dur(a_{t_{yes}}, (loc, \nu)) = \delta_{\Phi_{ft}^{\mathcal{T}}(t_{yes})} = \delta_0$,
  - $Dur(a_{t_{no}}, (loc, \nu)) = \delta_{\Phi_{ft}^{\mathcal{T}}(t_{no})} = \delta_0$, and
  - $Dur(a_{t_{remove}}, (loc, \nu)) = \delta_{\Phi_{ft}^{\mathcal{T}}(t_{remove})} = \delta_3$

  for $\nu \in \mathcal{V}^{\mathcal{H}}$.
- $Wgt(jump_{t_{random}}) = Wgt(jump_{t_{yes}}) = Wgt(jump_{t_{no}}) = Wgt(jump_{t_{remove}}) = 1$.

A visual representation of $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}_2}$ is given in Figure 4.3.

4.3. Transforming Continuous Fragments

We proceed with transforming the continuous part of the HPnG. For this, we first translate further relevant predicates from the semantics of HPnGs to the framework of the transformation. We start with the actual firing rate of continuous transitions

analogous to Definition 3.8.

Definition 4.5. Actual Firing Rate of Continuous Transitions

Let $\mathcal{H}$ be a HPnG and $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation of the variables. We define the *actual firing rate* of a continuous transition $t \in \mathcal{T}^{cont}$ as $rate_\nu \colon \mathcal{T}^{cont} \to \mathbb{R}_{\geq 0}$ with

$$rate_\nu(t) = \begin{cases} 0 & \text{if } \neg conc_\nu(t) \\ \Phi_{fr}^{\mathcal{T}}(t) \cdot \min\left\{\nu(l_{t,p}) \mid p \in \mathcal{P}^{cont}\right\} & \text{if } conc_\nu(t) \end{cases}$$

In comparison to Definition 3.8, we use $\nu(l_{t,p})$ instead of $l(t)(p)$ as before and additionally distinguish between t having concession or not in contrast to t being fireable or not. This does not make a difference since these notions coincide for continuous transitions by definition.

For describing the evolution of the markings in continuous places, we define the drift analogously to Definition 3.12. Again, both definitions coincide with the only difference that the property is defined for a valuation over the variables $\mathcal{V}^{\mathcal{H}}$ rather than for a state of $\mathcal{H}$.

Definition 4.6. Drift of a Continuous Place

Let $\mathcal{H}$ be a HPnG and $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation of the variables. The *drift* of the continuous places $\mathcal{P}^{cont}$ is defined as $drift_\nu \colon \mathcal{P}^{cont} \to \mathbb{R}$, where

$$drift_\nu(p) = \sum_{t_i \in \mathcal{I}^{cont}(p)} rate_\nu(t_i) - \sum_{t_j \in \mathcal{O}^{cont}(P)} rate_\nu(t_j).$$

For translating the rate adaption, we define functions applying the changes to the list of restrictions determined by the rate adaption algorithm analogous to Definitions 3.13 and 3.15. Also, we define a predicate expressing whether a place currently restricts a transition as was done in Definition 3.14.

Definition 4.7. Rate Reduction

Let $\mathcal{H}$ be a HPnG, $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation of the variables $Var^{\mathcal{H}}$, and $p \in \mathcal{P}^{cont}$ a continuous place. Further, let

$$in(p) = \sum_{t \in \mathcal{I}^{cont}(p)} rate_\nu(t)$$

to be the incoming flow to place p and

$$out(p) = \sum_{t \in \mathcal{O}^{cont}(p)} rate_\nu(t)$$

the outgoing flow to place p.

If $\nu(x_p) = 0$ and $drift_\nu(p) < 0$, we let $T = \{t \in \mathcal{O}^{cont}(p) \mid conc_\nu(t)\}$ be the outgoing fireable transitions of p and define

$$reduce_p^{empty}(\nu) \in \mathcal{V}^{\mathcal{H}}$$

to be

$$reduce_p^{empty}(\nu)(y) = \begin{cases} \frac{in(p)}{out(p)} \cdot \overbrace{\min\{\nu(l_{t,p'}) \mid p' \in \mathcal{P}^{cont}\}}^{\text{current strongest restriction}} & \text{if } y = l_{t,p} \text{ for } t \in T \\ \nu(y) & \text{else} \end{cases}$$

for all $y \in Var^{\mathcal{H}}$.

Dually, if $\nu(x_p) = \Phi_{ub}^{\mathcal{P}}(p)$ and $drift_\nu(p) > 0$, we let $T = \{t \in \mathcal{I}^{cont}(p) \mid conc_\nu(t)\}$ be the incoming fireable transitions of p and define

$$reduce_p^{full}(\nu) \in \mathcal{V}^{\mathcal{H}}$$

to be

$$reduce_p^{full}(\nu)(y) = \begin{cases} \frac{out(p)}{in(p)} \cdot \overbrace{\min\{\nu(l_{t,p'}) \mid p' \in \mathcal{P}^{cont}\}}^{\text{current strongest restriction}} & \text{if } y = l_{t,p} \text{ for } t \in T \\ \nu(y) & \text{else} \end{cases}$$

for all $y \in Var^{\mathcal{H}}$.

Definition 4.8. Placing Restrictions

Let $\mathcal{H}$ be a HPnG, $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation of the variables, and $p \in \mathcal{P}^{cont}$ a continuous place. We denote by $restr_\nu(p)$ whether p places restrictions on some transition in ν, i.e., $restr_\nu(p)$ is true iff there exists a transition $t \in \mathcal{I}^{cont}(p) \cup \mathcal{O}^{cont}(p)$ such that $\nu(l_{t,p}) < 1$.

Definition 4.9. Rate Reset

Let $\mathcal{H}$ be a HPnG, $\nu \in \mathcal{V}^{\mathcal{H}}$ a valuation of the variables, and $p \in \mathcal{P}^{cont}$ a continuous place such that $restr_\nu(p)$ holds, i.e., p currently restricts at least one incoming or outgoing transition.

If $\nu(x_p) = 0$ and $drift_\nu(p) > 0$ or $\nu(x_p) = \Phi_{ub}^{\mathcal{P}}(p)$ and $drift_\nu(p) < 0$, we define the valuation with updated restriction list $reset_p(l) \in \mathcal{V}^{\mathcal{H}}$ as

$$reset_p(\nu)(y) = \begin{cases} 1 & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{T}^{cont} \\ \nu(y) & \text{else} \end{cases}$$

for all $y \in Var^{\mathcal{H}}$.

As before, these definitions differ from the originals by referring to a valuation $\nu \in \mathcal{V}^{\mathcal{H}}$ instead of a state $\sigma \in \Sigma^{\mathcal{H}}$ and require continuous transitions to have concession instead of being fireable. Moreover, the $reduce(p)$ and the $reset(p)$ functions from Definitions 3.13 and 3.15 updated only the list of restrictions, while the functions from Definitions 4.7 and 4.9 update a full valuation, where all variables other than those representing the list of restrictions are unchanged. Using the translated predicates, we can define the SHA corresponding to a continuous place.

DEFINITION 4.10. **Transforming Continuous Places**

Let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG and $p \in \mathcal{P}^{cont}$ a continuous place. Then, we define a corresponding SHA $\mathcal{A}_p^{\mathcal{H}}$ as

$$\mathcal{A}_p^{\mathcal{H}} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

where

- $Loc = \{loc\}$.
- $Var = Var^{\mathcal{H}}$ with $Con = \{x_p\} \cup \{l_{t,p} \in L \mid t \in \mathcal{T}^{cont}\}$ and $NCon = Var^{\mathcal{H}} \backslash Con$.
- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- $Init(loc) = \{\nu \in \mathcal{V}^{\mathcal{H}} \mid \nu(x_p) = x_0(p) \wedge \forall t \in \mathcal{T}^{cont}.\ \nu(l_{t,p}) = 1\}$.
- $Edge = \{reduce_e, reduce_f, reset_e, reset_f, resetSingle\}$ where
  - $reduce_e = (loc, \mu_1, loc)$ with
$$\mu_1 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_p) = 0 \wedge drift_\nu(p) < 0 \\ \wedge\, \nu' = reduce_p^{empty}(\nu) \end{array} \right\},$$
  - $reduce_f = (loc, \mu_2, loc)$ with
$$\mu_2 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_p) = \Phi_{ub}^{\mathcal{P}}(p) \neq 0 \wedge drift_\nu(p) > 0 \\ \wedge\, \nu' = reduce_p^{full}(\nu) \end{array} \right\},$$
  - $reset_e = (loc, \mu_3, loc)$ with
$$\mu_3 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_p) = 0 \wedge restr_\nu(p) \wedge drift_\nu(p) > 0 \\ \wedge\, \nu' = reset_p(\nu) \end{array} \right\},$$

- $reset_f = (loc, \mu_4, loc)$ with

$$\mu_4 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_p) = \Phi^{\mathcal{P}}_{ub}(p) \wedge restr_\nu(p) \wedge drift_\nu(p) < 0 \\ \wedge\, \nu' = reset_p(\nu) \end{array} \right\},$$

- $resetSingle = (loc, \mu_5, loc)$ with

$$\mu_5 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \exists t \in \mathcal{T}^{cont}.\, \neg conc_\nu(t) \wedge \nu(l_{t,p}) < 1 \\ \wedge\, \nu' = \nu[l_{t,p} \mapsto 1] \end{array} \right\}.$$

- $Act(loc)$ is the set of activities that are solutions of

$$\dot{x}_p = drift_\nu(p) \quad \text{and} \quad \dot{l}_{t,p} = 0 \text{ for all } t \in \mathcal{T}^{cont}.$$

- $Lab = \{a_1, a_2, a_3, a_4, a_5\}$.
- $Proc$ is given by
  - $Proc(reduce_e) = a_1$,
  - $Proc(reduce_f) = a_2$,
  - $Proc(reset_e) = a_3$,
  - $Proc(reset_f) = a_4$, and
  - $Proc(resetSingle) = a_5$.
- For all $\nu \in \mathcal{V}^{\mathcal{H}}, a \in Lab$, we have $Dur(a, (loc, \nu)) = \delta_0$.
- $Wgt(e) = 1$ for all $e \in Edge$.

As the SHA corresponding to a discrete fragment, the SHA $\mathcal{A}^{\mathcal{H}}_p$ corresponding to a continuous place p is made of a single location loc. The controlled variables are x_p, representing the marking of p and $l_{t,p}$ representing the restriction posed on t by p for all continuous transitions t. Again, we do not place any conditions on the variables, i.e., the invariant contains all possible valuations. What we require from the variables is that x_p remains within its range, which is guaranteed by the rate adaption jumps, and that the restriction list entries $l_{t,p}$ are between zero and one, which is also ensured by definition. In the initial valuation, we require the marking for each continuous place to be set to the initial value specified in M_0 and the entries in the restriction list to be one, as p does initially not restrict any transitions.

There are five jumps from loc to loc. The first four correspond to the four cases of the rate adaption as defined in Algorithm 1. For example, the jump $reduce_e$ is enabled

when p is empty and has a negative drift, i.e., $\nu(x_p) = 0$ and $drift_\nu(p) < 0$. This is equivalent to the conditions of the first branch of Algorithm 1. When $reduce_e$ is taken, the valuation is updated according to Definition 4.7. This corresponds to the effects of the body of the first if-branch in Algorithm 1. The jump $reduce_f$ describes the second if-branch, i.e., the reduction of rates in case p is at an upper boundary. For resetting the rates, jumps $reset_e$ and $reset_f$ are defined. The fifth jump *resetSingle* makes sure that the restriction list is correct: If there is a continuous transition t without concession that is restricted by p, the corresponding variable $l_{t,p}$ is reset to one. In the activities, the derivative of the variable x_p is set to the drift of p, as the drift exactly describes how much fluid flows in and out of a place. The list restriction variables are not modified when time passes.

Since all defined jumps are *urgent*, meaning they have to be taken as soon as they are enabled, they are all assigned the Dirac distribution δ_0 via a unique label. If two jumps become enabled at the same time, we probabilistically decide which jump to take. Since we do not prefer any jump over another, we assign the same weight to each jump.

EXAMPLE 4.2. **Transforming Continuous Places**

We are going to continue with transforming the HPnG $\mathcal{H}$ discussed in Chapter 3. The discrete fragments were transformed in Example 4.1. There are two continuous places p_{plant} and p_{barr}. We start with constructing the SHA corresponding to p_{plant} as defined in Definition 4.10. We obtain

$$\mathcal{A}^{\mathcal{H}}_{p_{plant}} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

where

- $Loc = \{loc\}$.
- $Var = Var^{\mathcal{H}}$ with $Con = \left\{x_{p_{plant}}\right\} \cup \left\{l_{t,p_{plant}} \in L \;\middle|\; t \in \mathcal{T}^{cont}\right\}$
  $= \left\{x_{p_{plant}}, l_{t_{irr},p_{plant}}, l_{t_{rain},p_{plant}}, l_{t_{leak},p_{plant}}\right\}$
  and $NCon = Var^{\mathcal{H}} \setminus Con$.
- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- $Init(loc) = \left\{\nu \in \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(x_{p_{plant}}) = x_0(p_{plant}) \wedge \forall t \in \mathcal{T}^{cont}.\, \nu(l_{t,p_{plant}}) = 1\right\}$
  $= \left\{\nu \in \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(x_{p_{plant}}) = 0 \wedge \nu(l_{t,p_{plant}}) = 1\right\}$
  for $t \in \{t_{irr}, t_{rain}, t_{leak}\}$.
- $Edge = \{reduce_e, reduce_f, reset_e, reset_f, resetSingle\}$ where

– $reduce_e = (loc, \mu_1, loc)$ with

$$\mu_1 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{plant}}) = 0 \wedge drift_\nu(p_{plant}) < 0 \\ \wedge\, \nu' = reduce^{empty}_{p_{plant}}(\nu) \end{array} \right\}$$
$$= \emptyset,$$

– $reduce_f = (loc, \mu_2, loc)$ with

$$\mu_2 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{plant}}) = \Phi^{\mathcal{P}}_{ub}(p_{plant}) \wedge drift_\nu(p_{plant}) > 0 \\ \wedge\, \nu' = reduce^{full}_{p_{plant}}(\nu) \end{array} \right\}$$
$$= \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{plant}}) = 60 \wedge rate_\nu(t_{irr}) > 0 \\ \wedge\, \nu' = reduce^{full}_{p_{plant}}(\nu) \end{array} \right\}$$

– $reset_e = (loc, \mu_3, loc)$ with

$$\mu_3 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{plant}}) = 0 \wedge restr_\nu(p_{plant}) \\ \wedge\, drift_\nu(p_{plant}) > 0 \wedge \nu' = reset_{p_{plant}}(\nu) \end{array} \right\}$$
$$= \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{plant}}) = 0 \wedge restr_\nu(p_{plant}) \\ \wedge\, rate_\nu(t_{irr}) > 0 \wedge \nu' = reset_{p_{plant}}(\nu) \end{array} \right\},$$

– $reset_f = (loc, \mu_4, loc)$ with

$$\mu_4 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{plant}}) = \Phi^{\mathcal{P}}_{ub}(p_{plant}) \wedge restr_\nu(p_{plant}) \\ \wedge\, drift_\nu(p_{plant}) < 0 \wedge \nu' = reset_{p_{plant}}(\nu) \end{array} \right\}$$
$$= \emptyset$$

– $resetSingle = (loc, \mu_5, loc)$ with

$$\mu_5 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \exists t \in \mathcal{T}^{cont}.\, \neg conc_\nu(t) \wedge \nu(l_{t,p_{plant}}) < 1 \\ \wedge\, \nu' = \nu[l_{t,p_{plant}} \mapsto 1] \end{array} \right\}$$
$$= \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \neg conc_\nu(t_{irr}) \wedge \nu(l_{t_{irr},p_{plant}}) < 1 \\ \wedge\, \nu' = \nu[l_{t_{irr},p_{plant}} \mapsto 1] \end{array} \right\}$$

- $Act(loc)$ is the set of activities that are solutions of

$$\dot{x}_{p_{plant}} = drift_\nu(p_{plant}) = rate_\nu(t_{irr}) \quad \text{and} \quad \dot{l}_{t,p_{plant}} = 0$$

for $t \in \{t_{irr}, t_{rain}, t_{leak}\}$.

- $Lab = \{a_1, a_2, a_3, a_4, a_5\}$.
- $Proc$ is given by
  - $Proc(reduce_e) = a_1$,
  - $Proc(reduce_f) = a_2$,
  - $Proc(reset_e) = a_3$,
  - $Proc(reset_f) = a_4$, and
  - $Proc(resetSingle) = a_5$.
- $Dur(a, (loc, v)) = \delta_0$ for all $v \in \mathcal{V}^{\mathcal{H}}, a \in Lab$.
- $Wgt(e) = 1$ for all $e \in Edge$.

A depiction of $\mathcal{A}^{\mathcal{H}}_{p_{plant}}$ is given in Figure 4.4. We can see that the location loc controls the variable $x_{p_{plant}}$ for the level of p_{plant} and one variable per continuous transition for the restriction list entries, i.e., $l_{t_{irr},p_{plant}}$, $l_{t_{train},p_{plant}}$, and $l_{t_{leak},p_{plant}}$. The restriction list entries must be between zero and one. In the initial state, $x_{p_{plant}}$ must be zero and the place must not restrict any transition.

The jumps $reduce_e$ and $reset_f$ both require that p_{plant} has a negative drift. However, since p_{plant} does not have any outgoing transitions, the drift of this place cannot be negative. For this reason, we can essentially disregard both $reduce_e$ and $reset_f$ and the set of valuation pairs is empty. The jump $reduce_f$ restricts the incoming transition t_{irr} if p_{plant} is full, i.e., holds 60 liters, and has a positive drift. The drift is equal to the firing rate of t_{irr}, as this is the only transition connected to p_{plant}. The jump $reset_e$ removes the restrictions set by p_{plant} if the place is empty with a positive drift and currently restricts at least one transition. Note that in a valid path of this HPnG, this jump will never be taken as we only set restrictions when p_{plant} is full. The set of activities specify that the level of p_{plant} changes according to the drift, which is equal to the actual firing rate of t_{irr}.

Every jump is urgent, so we assign Dirac distribution to every label. Additionally, as motivated above, each jump is assigned a weight of one, because we do not prefer one jump over another.

For the second continuous place p_{barr}, we obtain

$$\mathcal{A}^{\mathcal{H}}_{p_{barr}} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt),$$

where

- $Loc = \{loc\}$
- $Var = Var^{\mathcal{H}}$ with $Con = \{x_{p_{barr}}, l_{t_{train},p_{barr}}, l_{t_{leak},p_{barr}}, l_{t_{irr},p_{barr}}\}$ and $NCon = Var \backslash Con$.

- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- $Init(loc) = \left\{ \nu \in \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(x_{p_{barr}}) = 50 \wedge \nu(l_{t,p_{barr}}) = 1 \right\}$ for $t \in \{t_{irr}, t_{rain}, t_{leak}\}$.
- $Edge = \{reduce_e, reduce_f, reset_e, reset_f, resetSingle\}$ where
  - $reduce_e = (loc, \mu_1, loc)$ with
$$\mu_1 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{barr}}) = 0 \wedge drift_\nu(p_{barr}) < 0 \\ \wedge\, \nu' = reduce^{empty}_{p_{barr}}(\nu) \end{array} \right\},$$
  - $reduce_f = (loc, \mu_2, loc)$ with
$$\mu_2 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{barr}}) = 100 \wedge drift_\nu(p_{barr}) > 0 \\ \wedge\, \nu' = reduce^{full}_{p_{barr}}(\nu) \end{array} \right\},$$
  - $reset_e = (loc, \mu_3, loc)$ with
$$\mu_3 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{barr}}) = 0 \wedge restr_\nu(p_{barr}) \\ \wedge\, drift_\nu(p_{barr}) > 0 \wedge \nu' = reset_{p_{barr}}(\nu) \end{array} \right\},$$
  - $reset_f = (loc, \mu_4, loc)$ with
$$\mu_4 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \nu(x_{p_{barr}}) = 100 \wedge restr_\nu(p_{barr}) \\ \wedge\, drift_\nu(p_{barr}) < 0 \wedge \nu' = reset_{p_{barr}}(\nu) \end{array} \right\},$$
  - $resetSingle = (loc, \mu_5, loc)$ with
$$\mu_5 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \begin{array}{l} \exists t \in \mathcal{T}^{cont}.\, \neg conc_\nu(t) \wedge \nu(l_{t,p_{barr}}) < 1 \\ \wedge\, \nu' = \nu[l_{t,p_{barr}} \mapsto 1] \end{array} \right\},$$
- $Act(loc)$ is the set of activities that are solutions of
$$\dot{x}_{p_{barr}} = drift_\nu(p_{barr}) \quad \text{and} \quad \dot{l}_{t,p_{barr}} = 0$$
for $t \in \{t_{irr}, t_{rain}, t_{leak}\}$.
- $Lab = \{a_1, a_2, a_3, a_4, a_5\}$
- $Proc$ is given by
  - $Proc(reduce_e) = a_1$,

  - $Proc(reduce_f) = a_2$,
  - $Proc(reset_e) = a_3$,
  - $Proc(reset_f) = a_4$, and
  - $Proc(resetSingle) = a_5$.
- $Dur(a, (loc, v)) = \delta_0$ for all $v \in \mathcal{V}^{\mathcal{H}}, a \in Lab$.
- $Wgt(e) = 1$ for all $e \in Edge$.

A depiction of $\mathcal{A}^{\mathcal{H}}_{p_{barr}}$ is given in Figure 4.5. Note that, in contrast to p_{plant}, the place p_{barr} has both incoming and outgoing transitions. Thus the set of valuation pairs is non-empty for every jump.

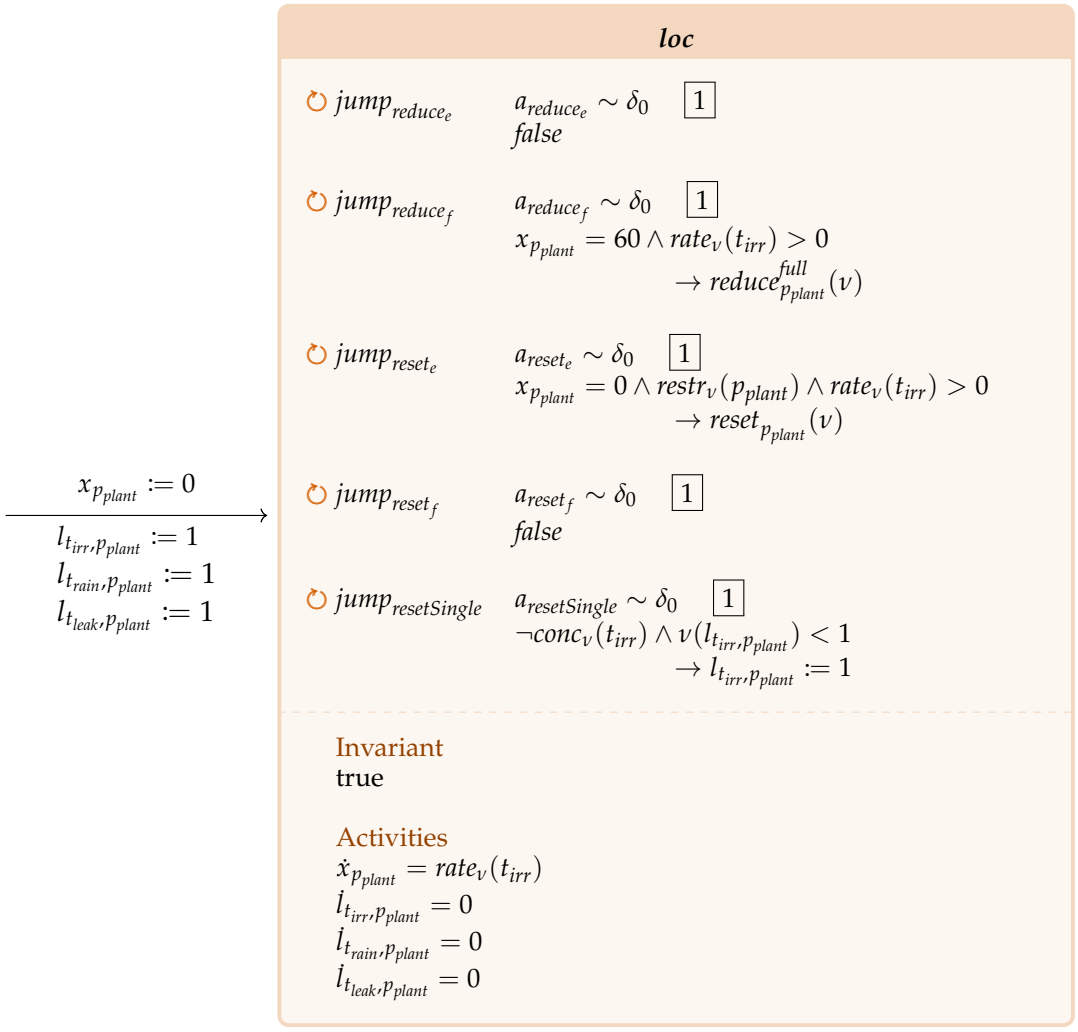


Figure 4.4.: Depiction of the SHA $\mathcal{A}^{\mathcal{H}}_{p_{plant}}$ discussed in Example 4.2.

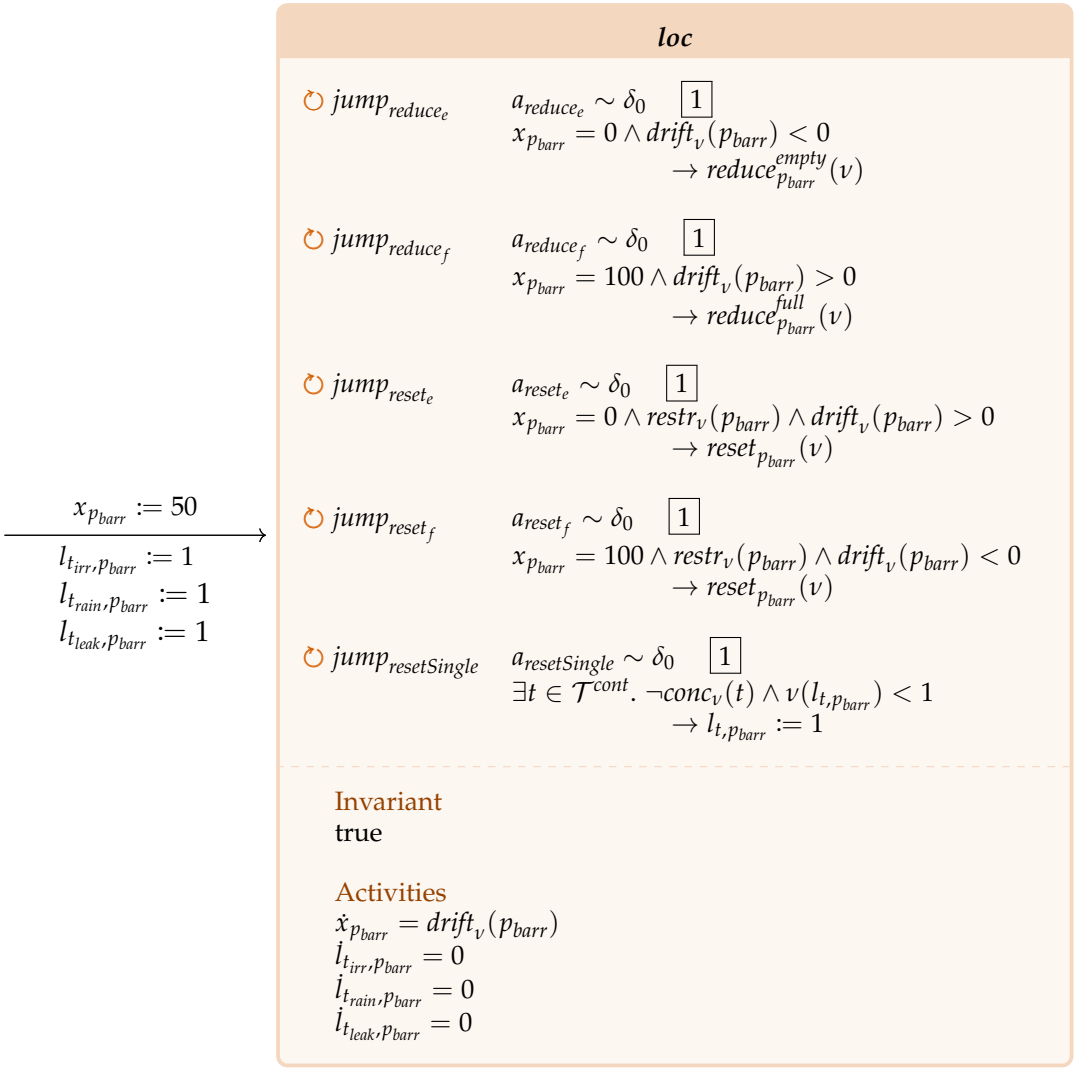


Figure 4.5.: Depiction of the SHA $\mathcal{A}^{\mathcal{H}}_{p_{barr}}$ discussed in Example 4.2.

4.4. Compositional Transformation

In the main theorem of this thesis, we combine the findings and definitions from the previous sections of this chapter. As discussed before, we obtain the complete transformation of a HPnG by composing the transformations of all discrete fragments and continuous places.

> DEFINITION 4.11. **Transforming HPnGs to SHAs**
> Let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG with discrete fragments $\mathcal{D}_1, \ldots, \mathcal{D}_n$ and continuous places $p_1, p_2, \ldots, p_m \in \mathcal{P}^{cont}$. Then, we define the transformation of $\mathcal{H}$ to an SHA $\mathcal{A}^{\mathcal{H}}$ as
>
> $$\mathcal{A}^{\mathcal{H}} = \mathcal{A}^{\mathcal{H}}_{\mathcal{D}_1} \parallel \mathcal{A}^{\mathcal{H}}_{\mathcal{D}_2} \parallel \cdots \parallel \mathcal{A}^{\mathcal{H}}_{\mathcal{D}_n} \parallel \mathcal{A}^{\mathcal{H}}_{p_1} \parallel \mathcal{A}^{\mathcal{H}}_{p_2} \parallel \cdots \parallel \mathcal{A}^{\mathcal{H}}_{p_m}.$$

Note that the composition as such is well defined as each variable is controlled by exactly one of the composed SHAs. $\mathcal{A}^{\mathcal{H}}$ combines the behavior of all sub-SHAs. We refer to $\mathcal{A}^{\mathcal{H}}_{\mathcal{D}}$ as the sub-SHA corresponding to a discrete fragment $\mathcal{D}$ and dually to $\mathcal{A}^{\mathcal{H}}_{p}$ as the sub-SHA corresponding to a continuous place p. Let $\mathcal{H}$ be a HPnG and $\mathcal{A}^{\mathcal{H}}$ the transformed SHA with

$$\mathcal{A}^{\mathcal{H}} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt).$$

We now discuss how the components look in more detail.

- Since every sub-SHA is only made of one location, so is the composed SHA $\mathcal{A}^{\mathcal{H}}$. The location is again referred to as *loc*.
- The set of variables is $Var^{\mathcal{H}}$ as defined in Definition 4.1.
- The set of invariants $Inv(loc)$ contains all valuations $\nu \in \mathcal{V}^{\mathcal{H}}$.
- The initial condition $Init(loc)$ contains a valuation $\nu \in \mathcal{V}^{\mathcal{H}}$ iff
  - $\nu(m_p) = m_0(p)$ for all $p \in \mathcal{P}^{disc}$,
  - $\nu(x_p) = x_0(p)$ for all $p \in \mathcal{P}^{cont}$, and
  - $\nu(l_{t,p}) = 1$ for all $p \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont}$.
- The set of jumps consists of one jump for every discrete transition and five jumps for each continuous place:

$$Edge = \left\{ jump_t \;\middle|\; t \in \mathcal{T}^{disc} \right\} \cup \bigcup_{p \in \mathcal{P}^{cont}} \{reduce_e@p, reduce_f@p, reset_e@p, reset_f@p, resetSingle@p\}$$

We denote by $@p$ from which sub-SHA a jump is originating. If this is clear, we might omit the tag $@p$ for better readability.

- The set of activities is given by the functions satisfying
  - $\dot{m}_p = 0$ for all $p \in \mathcal{P}^{disc}$,
  - $\dot{x}_p = drift_v(p)$ for all $p \in \mathcal{P}^{cont}$, and
  - $\dot{l}_{t,p} = 0$ for all $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$.
- The set of labels is given by

$$Lab = \left\{ a_t \;\middle|\; t \in \mathcal{T}^{disc} \right\} \cup \left\{ a_i@p \;\middle|\; i \in \{1, \ldots, 5\}, p \in \mathcal{P}^{cont} \right\},$$

  where we again write $a@p$ for a label originating from the SHA corresponding to p and omit the tag if it is unnecessary.
- *Proc* is the straightforward extension of the functions from the sub-SHAs.
- The durations are also assigned straightforward:

$$Dur(a, (loc, v)) = \begin{cases} \Phi^{\mathcal{T}}_{gt}(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{gen} \\ \delta_{\Phi^{\mathcal{T}}_{ft}(t)} & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{det} \\ \delta_0 & \text{else} \end{cases}$$

  for all $v \in \mathcal{V}^{\mathcal{H}}$ and $a \in Lab$.
- The weights of the jumps are given by

$$Wgt(e) = \begin{cases} \Phi^{\mathcal{T}}_p(t) & \text{if } e = jump_t \\ 1 & \text{else} \end{cases}$$

  for $e \in Edge$.

Note that by assigning the weights in this way, we obtain the same extended method for resolving conflicts as was described in Section 3.2.3. Should the HPnG rely on another scheduler to solve its conflicts, this can be transformed to a scheduler solving conflicts in the transformed HPnG in a straightforward manner.

EXAMPLE 4.3. **Transformation**
We finish the transformation started in the previous examples Example 4.1 and Example 4.2 and let

$$\mathcal{A}_H = \mathcal{A}^{\mathcal{H}}_{\mathcal{D}_1} \parallel \mathcal{A}^{\mathcal{H}}_{\mathcal{D}_2} \parallel \mathcal{A}^{\mathcal{H}}_{p_{plant}} \parallel \mathcal{A}^{\mathcal{H}}_{p_{barr}}.$$

More precisely, we have

$$\mathcal{A}^{\mathcal{H}} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt)$$

where

- $Loc = \{loc\}$.
- The set of variables is $Var^{\mathcal{H}} = M \cup X \cup L$ with
  - $M = \{m_{p_{rainmaker}}, m_{p_{control}}, m_{p_{choice}}\}$,
  - $X = \{x_{p_{barr}}, x_{p_{plant}}\}$, and
  - $L = \{l_{t_{rain},p_{barr}}, l_{t_{leak},p_{barr}}, l_{t_{irr},p_{barr}}, l_{t_{rain},p_{plant}}, l_{t_{leak},p_{plant}}, l_{t_{irr},p_{plant}}\}$

  as defined in Definition 4.1.
- $Inv(loc) = \mathcal{V}^{\mathcal{H}}$.
- A valuation ν is initial, i.e., $\nu \in Init(loc)$, if it fulfills the following conditions:
  - $\nu(m_{p_{control}}) = \nu(m_{p_{rainmaker}}) = 1$ and $\nu(m_{p_{choice}}) = 0$,
  - $\nu(x_{p_{barr}}) = 50, \nu(x_{p_{plant}}) = 0$, and
  - $\nu(l_{t,p_{barr}}) = \nu(l_{t,p_{plant}}) = 1$

  for $t \in \{t_{irr}, t_{rain}, t_{leak}\}$.
- The set of jumps consists of one jump for every discrete transition, originating from the discrete fragments, and five jumps for each continuous place:

$$
\begin{aligned}
&Edge \\
&= \{jump_{t_{on}}, jump_{t_{off}}\} \\
&\cup \{jump_{t_{random}}, jump_{t_{yes}}, jump_{t_{no}}, jump_{t_{remove}}\} \\
&\cup \{reduce_e@p_{barr}, reduce_f@p_{barr}, reset_e@p_{barr}, reset_f@p_{barr}, resetSingle@p_{barr}\} \\
&\cup \{reduce_e@p_{plant}, reduce_f@p_{plant}, reset_e@p_{plant}, reset_f@p_{plant}, resetSingle@p_{plant}\}
\end{aligned}
$$

  The guards and effects of the jumps are exactly as in the sub-SHAs.
- The set of activity functions is given by the functions satisfying
  - $\dot{m}_{p_{rainmaker}} = \dot{m}_{p_{control}} = \dot{m}_{p_{choice}} = 0$,
  - $\dot{x}_{p_{barr}} = drift_\nu(p_{barr})$ and $\dot{x}_{p_{plant}} = drift_\nu(p_{plant})$, and
  - $\dot{l}_{t,p_{barr}} = \dot{l}_{t,p_{plant}} = 0$

  for $t \in \{t_{irr}, t_{rain}, t_{leak}\}$.

- The set of labels is given by

$$\begin{aligned} Lab = &\left\{a_{t_{on}}, a_{t_{off}}\right\} \cup \left\{a_{t_{random}}, a_{t_{yes}}, a_{t_{no}}, a_{t_{remove}}\right\} \\ &\cup \left\{a_1@p_{barr}, a_2@p_{barr}, a_3@p_{barr}, a_4@p_{barr}, a_5@p_{barr}\right\} \\ &\cup \left\{a_1@p_{plant}, a_2@p_{plant}, a_3@p_{plant}, a_4@p_{plant}, a_5@p_{plant}\right\} \end{aligned}$$

- *Proc* is the straightforward extension of the functions from the sub-SHAs.
- *Dur* also combines the previous definitions, and we obtain:

$$Dur(a,(loc,v)) = \begin{cases} exp & \text{if } a = a_{t_{on}} \text{ or } a = a_{t_{off}} \\ \mathcal{U}_{[1,10]} & \text{if } a = a_{t_{random}} \\ \delta_3 & \text{if } a = a_{t_{remove}} \\ \delta_0 & \text{else} \end{cases}$$

  for $v \in \mathcal{V}^{\mathcal{H}}$ and $a \in Lab$.
- The weights of the jumps are given by $Wgt(e) = 1$ for all $e \in Edge$.

A visualization of the SHA $\mathcal{A}^{\mathcal{H}}$ is given in Figure 4.6.

Comparing the original HPnG $\mathcal{H}$ to its transformation $\mathcal{A}^{\mathcal{H}}$ as in Example 4.3, we can see that the structure of the system is lost, and it is difficult to recognize what the SHA is simulating. Therefore, when modeling systems, it can be helpful to model a system as a HPnG and then (in the best case automatically) transform the HPnG to an SHA.

Classifying the Transformed HPnG. Pilch et al. showed in [Pil+20] that bounded executions of HPnGs can be modeled by SHAs of a certain subclass, namely by *singular automata extended with random clocks*. Essentially, these are based on a restricted class of hybrid automata, referred to as *rectangular automata*, with an additional mechanism for clocks. A *rectangular automaton* is a hybrid automaton where the set of initial valuations, invariants, activities, as well as guards and effects of jumps are described by *rectangular* sets, meaning that each set can be written as a Cartesian product of intervals with rational or infinite endpoints. In particular, this has the consequence that the conditions placed on every variable may not depend on the valuation of other variables. If additionally the set of activities for a location is given as a singleton for each variable, i.e., the derivatives are constant, we speak of a *singular automaton*. For modeling discrete transitions, Pilch et al. extend singular automata with random clocks to control when a jump is fireable. This is very similar to the stochastic component of the SHAs defined in this thesis.

In the transformation given in Definition 4.11, the sets of initial valuations and invariants are rectangular. However, the guards of the jump cannot be described by rectangular sets. For example, consider the jump $reset_e$ originating from the SHA of a continuous place p. The guard of $reset_e$ requires that $restr_v(p)$ holds, i.e., that there exists a transition that is currently restricted by p. This cannot be expressed using rectangular sets, as the allowed values for a restriction list variable depends on the current valuation of the other restriction list variables. Therefore, the result of our transformation syntactically does not fit into the category of singular automata extended with random clocks.

States of the Transformation. Now, we take a look at the state set of $\mathcal{A}^{\mathcal{H}}$ given by

$$\Theta^{\mathcal{A}^{\mathcal{H}}} = \{loc\} \times \mathcal{V} \times \mathcal{V}_{Lab}.$$

A state is made of the location loc, a valuation over the variables from Definition 4.1 and a valuation over all labels. For the latter, note that by definition of the transformation, only the values for labels a_t for $t \in \mathcal{T}^{disc}$ are potentially non-zero. All others are zero, at least in every reachable valuation, as their corresponding jumps are urgent.

EXAMPLE 4.4. **States and Paths in Transformed HPnG**

Consider the transformed HPnG $\mathcal{A}^{\mathcal{H}}$ from Example 4.3. In Example 3.8, we discussed the prefix of a path in the HPnG $\mathcal{H}$. Now, we will see that a corresponding prefix exists in $\mathcal{A}^{\mathcal{H}}$. We have that

$$\vartheta_0 \xrightarrow{\tau = 1} \vartheta_1 \xrightarrow{\text{fire } t_{on}} \vartheta_2 \xrightarrow{\tau = 2} \vartheta_3,$$

where the formal definition of each state ϑ_i is given in Table 4.1. We again do not explicitly give the activation function as it is unique for the location loc.

We start in ϑ_0. This is an initial state, because the markings of the discrete and continuous places match the values specified in Example 4.3. Further, the sampled values for the clock are in the support of the corresponding distributions. A more detailed argumentation was provided in Example 3.1. One can easily see that the arguments transfer.

To get to ϑ_1, we let one time unit pass. Note that this is possible because no jump is fireable. Even though some clocks have run out, for example the one corresponding to t_{yes}, the respective jumps are not enabled and consequently not fireable. Due to the time step, $x_{p_{barr}}$ is decreased by one, because the current drift is minus one. This is because the rate of t_{leak} is one, while all other continuous transitions have a current rate of one. The level of p_{plant} does not change as the drift is zero. Additionally, the clocks of t_{on}, t_{random}, and t_{remove} decreased. This is because the corresponding jumps are enabled.

In ϑ_1, the jump t_{on} is enabled and its clock has run out. Thus, we have to take the

jump, which modifies the markings by increasing $x_{p_{rainmaker}}$ by one, i.e., adding a token to $p_{rainmaker}$ in ϑ_2.

Then, we let two time units pass and reach ϑ_3. The drift of p_{barr} now is nine, because in contrast to ϑ_0, the actual rate of the continuous transition t_{rain} is ten. The clock values are decreased by two, this time including the label for t_{off}.

This path prefix coincides with the one discussed in Example 3.8, and we can already forebode how the states of the two systems relate. We will give a formal definition of this connection in the next section.

Possible Extensions. The transformation as defined above does not specifically require a HPnG to be acyclic. Neither do we require a transition to have only one incoming and one outgoing arc. Therefore, the transformation is directly applicable to cyclic HPnGs as well as to HPnGs containing transitions with multiple incoming or outgoing arcs. We imposed this restriction mainly because of the rate adaption, as we can not guarantee termination for such HPnGs.

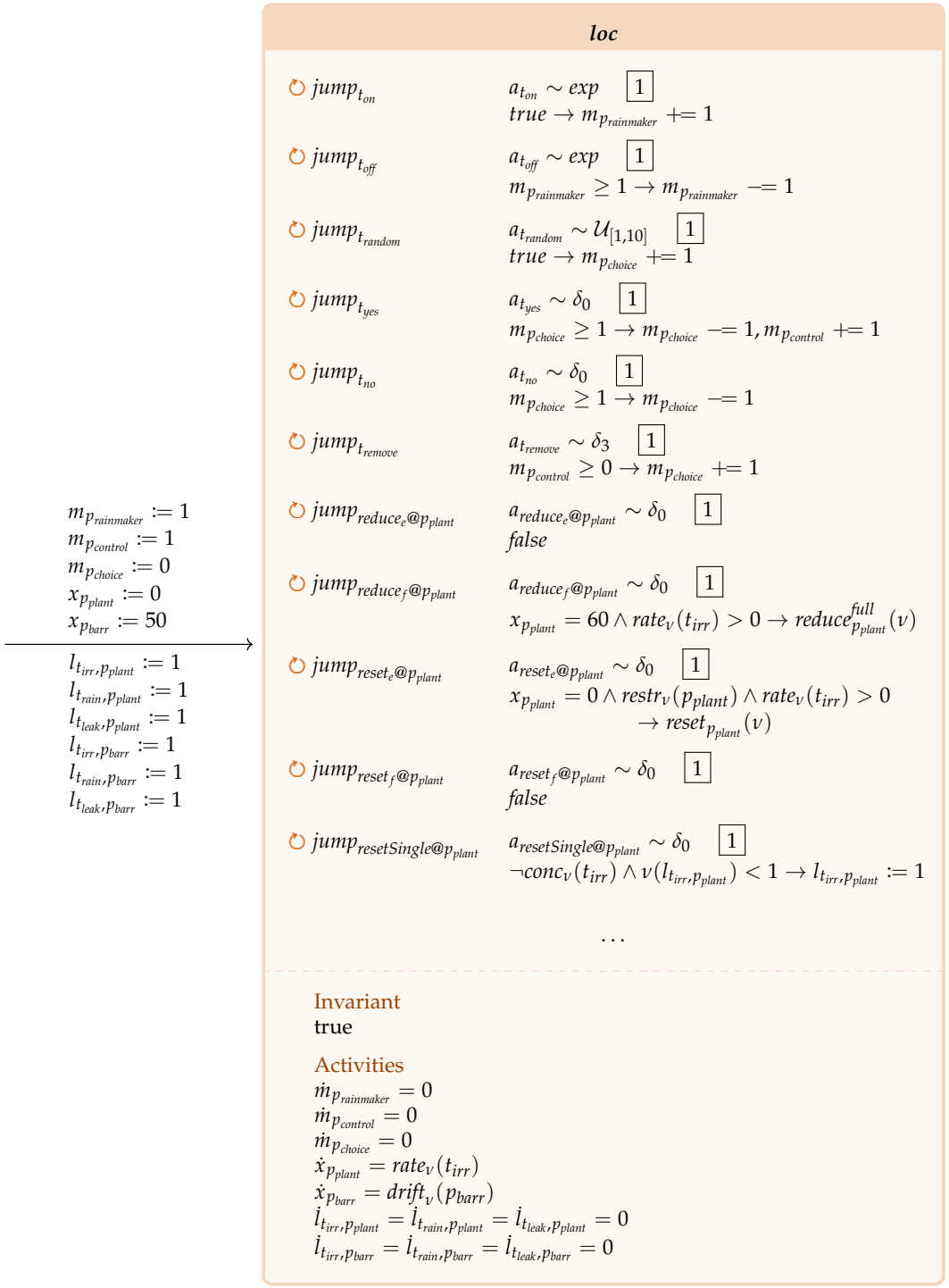


Figure 4.6.: A depiction of the SHA $\mathcal{A}^{\mathcal{H}}$ modeling the full irrigation system as depicted in Figure 4.1. The components are discussed in Example 4.2 in more detail. Due to space constraints, we omit five jumps, including for example $jump_{reduce_e@p_{barr}}$ originating from p_{barr}.

State	Location	Variables ν		Labels ν_{Lab}	
ϑ_0	loc	$m_{p_{rainmaker}}$	$\mapsto 1$	$a_{t_{on}}$	$\mapsto 1$
		$m_{p_{control}}$	$\mapsto 1$	$a_{t_{off}}$	$\mapsto 2$
		$m_{p_{choice}}$	$\mapsto 0$	$a_{t_{random}}$	$\mapsto 5$
		$x_{p_{barr}}$	$\mapsto 50$	$a_{t_{yes}}$	$\mapsto 0$
		$x_{p_{plant}}$	$\mapsto 0$	$a_{t_{no}}$	$\mapsto 0$
		$l_{t,p}$	$\mapsto 1$	$a_{t_{remove}}$	$\mapsto 3$
				a	$\mapsto 0$ for all other labels
ϑ_1	loc	$m_{p_{rainmaker}}$	$\mapsto 1$	$a_{t_{on}}$	$\mapsto \mathbf{0}$
		$m_{p_{control}}$	$\mapsto 1$	$a_{t_{off}}$	$\mapsto 2$
		$m_{p_{choice}}$	$\mapsto 0$	$a_{t_{random}}$	$\mapsto \mathbf{4}$
		$x_{p_{barr}}$	$\mapsto \mathbf{49}$	$a_{t_{yes}}$	$\mapsto 0$
		$x_{p_{plant}}$	$\mapsto 0$	$a_{t_{no}}$	$\mapsto 0$
		$l_{t,p}$	$\mapsto 1$	$a_{t_{remove}}$	$\mapsto \mathbf{3}$
				a	$\mapsto 0$ for all other labels
ϑ_2	loc	$m_{p_{rainmaker}}$	$\mapsto \mathbf{2}$	$a_{t_{on}}$	$\mapsto \mathbf{3}$
		$m_{p_{control}}$	$\mapsto 1$	$a_{t_{off}}$	$\mapsto 2$
		$m_{p_{choice}}$	$\mapsto 0$	$a_{t_{random}}$	$\mapsto 4$
		$x_{p_{barr}}$	$\mapsto 49$	$a_{t_{yes}}$	$\mapsto 0$
		$x_{p_{plant}}$	$\mapsto 0$	$a_{t_{no}}$	$\mapsto 0$
		$l_{t,p}$	$\mapsto 1$	$a_{t_{remove}}$	$\mapsto 3$
				a	$\mapsto 0$ for all other labels
ϑ_2	loc	$m_{p_{rainmaker}}$	$\mapsto 2$	$a_{t_{on}}$	$\mapsto \mathbf{1}$
		$m_{p_{control}}$	$\mapsto 1$	$a_{t_{off}}$	$\mapsto \mathbf{0}$
		$m_{p_{choice}}$	$\mapsto 0$	$a_{t_{random}}$	$\mapsto \mathbf{2}$
		$x_{p_{barr}}$	$\mapsto \mathbf{67}$	$a_{t_{yes}}$	$\mapsto 0$
		$x_{p_{plant}}$	$\mapsto 0$	$a_{t_{no}}$	$\mapsto 0$
		$l_{t,p}$	$\mapsto 1$	$a_{t_{remove}}$	$\mapsto \mathbf{1}$
				a	$\mapsto 0$ for all other labels

Table 4.1.: A tabular representation of a path prefix of $\mathcal{A}^{\mathcal{H}}$ from Example 4.3. Since the list of restriction does not change, we write $l_{t,p}$ representing every entry in the restriction list. Modified values are highlighted. The semantic steps taken to get from ϑ_i to ϑ_{i+1} are discussed in Example 4.4.

4.5. Correctness of the Transformation

At the beginning of this chapter, we mentioned that the notion of time differs in HPnGs and SHAs. We will briefly justify how this is handled in our transformation. Note that the discrepancies only emerge when the invariants in a location are violated and no jump leaving the location is possible. Then, time stops in an SHA. In contrast, time always continues to pass in HPnGs. In the transformation, we only use the invariant *true*, which can never be violated. Thus, time can always continue to pass in the transformed HPnG, matching its original behavior. We will now prove that the original HPnG and its transformed SHA are semantically equivalent.

THEOREM 4.1. **Correctness of the Transformation**
The transformation of HPnGs to SHAs as defined in Definition 4.11 is correct, in the sense that a HPnG $\mathcal{H}$ is semantically equivalent to its transformation $\mathcal{A}^{\mathcal{H}}$. We can give a function $\tilde{f}\colon \Sigma^{\mathcal{H}} \to \Theta^{\mathcal{A}^{\mathcal{H}}}$ such that

$$\sigma \Rightarrow \sigma' \text{ iff } \tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$$

for all $\sigma, \sigma' \in \Sigma^{\mathcal{H}}$.

Proof. Let $\mathcal{H}$ be a HPnG and $\mathcal{A}^{\mathcal{H}}$ the SHA resulting from the transformation. We define a function $\tilde{f}\colon \Sigma^{\mathcal{H}} \to \Theta^{\mathcal{A}^{\mathcal{H}}}$ mapping states of $\mathcal{H}$ to states of $\mathcal{A}^{\mathcal{H}}$ as $\tilde{f}(m, x, c, l) = (loc, \nu, \nu_{Lab})$, with ν and ν_{Lab} defined as follows:

- Recall that the set of variables is defined as $Var^{\mathcal{H}} = M \cup X \cup L$. We define

$$\nu(y) = \begin{cases} m(p) & \text{if } y = m_p \in M \text{ for } p \in \mathcal{P}^{disc} \\ x(p) & \text{if } y = x_p \in X \text{ for } p \in \mathcal{P}^{cont} \\ l(t)(p) & \text{if } y = l_{t,p} \in L \text{ for } t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont} \end{cases}$$

  for all $y \in Var^{\mathcal{H}}$.

- We use ν_{Lab} to model the clocks of $\mathcal{H}$ and define

$$\nu_{Lab}(a) = \begin{cases} c(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{disc} \\ 0 & \text{else} \end{cases}$$

  for all $a \in Lab$.

This matches the purpose of the defined variables, and the intuition of the connection formulated in Example 4.4: The finite prefix of a path

$$\vartheta_0 \Rightarrow \vartheta_1 \Rightarrow \vartheta_2 \Rightarrow \vartheta_3$$

discussed in the example corresponds to

$$\sigma_0 \Rightarrow \sigma_1 \Rightarrow \sigma_2 \Rightarrow \sigma_3$$

from Example 3.8, and we can easily verify that $\tilde{f}(\sigma_i) = \vartheta_i$ for $i \in \{0, \ldots, 3\}$.

The inverse function $\tilde{f}^{-1}$ maps states of $\mathcal{A}^{\mathcal{H}}$ to states of $\mathcal{H}$ and is given by

$$\tilde{f}^{-1}(loc, \nu, \nu_{Lab}) = (m, x, c, l), \text{ where}$$

- $m(p) = \nu(m_p)$ for $p \in \mathcal{P}^{disc}$,
- $x(p) = \nu(x_p)$ for $p \in \mathcal{P}^{cont}$,
- $c(t) = \nu_{Lab}(a_t)$ for $t \in \mathcal{T}^{disc}$, and
- $l(t)(p) = \nu(l_{t,p})$ for $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$.

We can prove that $\tilde{f}^{-1}$ is the inverse of $\tilde{f}$ by verifying that $\tilde{f}(\tilde{f}^{-1}((loc, \nu, \nu_{Lab}))) = (loc, \nu, \nu_{Lab})$ and $\tilde{f}^{-1}(\tilde{f}(m, x, c, l)) = (m, x, c, l)$. The relation induced by $\tilde{f}$, more precisely $\{(\sigma, \tilde{f}(\sigma)) \mid \sigma \in \Sigma^{\mathcal{H}}\}$, is a bisimulation. Under abuse of notation, we sometimes also call $\tilde{f}$ a bisimulation.

We start the proof of correctness by showing some properties of the defined predicates. Basically, we validate that the predicates defined in this chapter correspond to those defined for HPnGs, and thereby confirm that we defined them in a meaningful way.

THEOREM 4.2. **Correctness of the Predicates**

Let $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$ be a HPnG, $\sigma \in \Sigma^{\mathcal{H}}$ and $\tilde{f}(\sigma) = (l, \nu, \nu_{Lab}) \in \Theta^{\mathcal{A}^{\mathcal{H}}}$. Then, it holds that

(1) $conc_\sigma(t) \iff conc_\nu(t)$ for $t \in \mathcal{T}$

(2) $conc_\sigma(t) \iff enabled_{\tilde{f}(\sigma)}(jump_t)$ for $t \in \mathcal{T}^{disc}$

(3) $fireable_\sigma(t) \iff fireable_{\tilde{f}(\sigma)}(jump_t)$ for $t \in \mathcal{T}^{disc}$

(4) $restr_\sigma(p) \iff restr_\nu(p)$ for $p \in \mathcal{P}^{cont}$

(5) $rate_\sigma(t) = rate_\nu(t)$ for $t \in \mathcal{T}^{cont}$

(6) $drift_\sigma = drift_\nu$

These claims follow straightforward from the definition of the predicates and $\tilde{f}$. A detailed proof is given in Appendix B.1.

Using Theorem 4.2, we can prove the correctness of the transformation by induction. For this, we first show that the set of initial states coincides with respect to the bisimulation $\tilde{f}$, and then prove that $\sigma \Rightarrow \sigma'$ iff $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$. Details can be found in Appendix B.2. □

Reachability Probabilities. Since we consider probabilistic systems, we need to ensure that states of the HPnG and corresponding states in the transformation are reached with the same probability. Thus, it is not only of interest that each path of the HPnG has a corresponding path in the transformation, but also that these paths have the same probabilities. We argue that this is indeed the case for the transformation defined above. Stochasticity is introduced in two ways: First, when sampling the firing times for general transitions, and second when solving conflicts between several applicable semantic steps. The firing times of a general transition corresponds to the enabling duration of the jump that models the firing of the respective transition. By definition, these are sampled from the same distribution. When several steps are possible, priorities are used to resolve conflicts in the HPnG as described in Section 3.2.3. In an SHA, such conflicts are solved by a probabilistic choice based on the weights as described in Section 2.2. The weights of jumps in the transformation are equal to the priorities in the HPnG by definition. Consequently, the probabilities are also in agreement here. In conclusion, the uncertainties along equivalent paths of the HPnG and its transformation are based on the same distributions and therefore the respective states are reached with the same probabilities.

A Comment on Rate Adaption Termination. In Section 3.2.2, we only proved termination of the rate adaption algorithm for certain subclasses of HPnGs. Now that we have shown the equivalence of the HPnG $\mathcal{H}$ and its transformation $\mathcal{A}^{\mathcal{H}}$, we know in particular that the rate adaption behaves equally in $\mathcal{H}$ and $\mathcal{A}^{\mathcal{H}}$. In particular, if the rate adaption does not terminate in a state σ of $\mathcal{H}$, we know that an equal non-terminating path exists that starts in $\tilde{f}(\sigma)$, where $\tilde{f}$ is the bisimulation function defined in the proof of Theorem 4.1. If we can prove the termination of the rate adaption for a larger class of HPnGs as formulated in Conjecture 3.1, this result transfers to their transformations.

5. Conclusion

In this thesis, we provided a formal transformation from HPnGs to SHAs, including a proof of correctness. We started in Chapter 2 by defining a modeling language for SHAs introduced in recent work by Gerlach [Ger22]. Building on the definition of HPnGs by Gribaudo and Remke [GR16], we defined the semantics of HPnGs using a set of inference rules and presented a novel algorithm for adapting the rates of continuous transitions in Chapter 3. We proved the termination of the algorithm for a subclass of HPnGs. In Chapter 4, we defined a formal transformation of HPnGs to SHAs and proved the correctness using a bisimulation.

5.1. Future Work

In Section 3.2.2, we presented an algorithm for rate adaption and proved its termination for a subclass of HPnGs, namely for strongly acyclic HPnGs. Moreover, we proved that the algorithm terminates in all acyclic HPnGs starting in a state where all critical places are empty (or dually, all critical places are full). However, as formulated in Conjecture 3.1, we strongly believe that the algorithm terminates for all acyclic HPnGs. This is to be proven in future work. Some ideas and attempts for the proof are already outlined in Section 3.2.2.

On a larger scale, it would be worthwhile to study real-world examples such as the waste water treatment facility discussed and analyzed by Ghasemieh et al. [GRH16]. To test this efficiently, an automation of the presented transformation is of interest. This could be implemented using the tool `HYPEG`[1] [PER17], which provides a library for simulating HPnGs, or `HPnmG`[2] [HNR20], which is a tool for model checking HPnGs.

In the process of the implementation, possible optimizations of the transformation could also be investigated. For example, we saw in Example 4.2 that some jumps can be omitted if a continuous place has no outgoing arcs. Similarily, several jumps become obsolete when a place has no finite upper bound. Combining such findings could reduce the size of the resulting SHA and thus make computations more efficient.

[1] `https://zivgitlab.uni-muenster.de/ag-sks/tools/hypeg`
[2] `https://zivgitlab.uni-muenster.de/ag-sks/tools/hpnmg`

A. Notations

In the following, we list notations used in this thesis sorted by first occurrence per chapter. The lists are not intended to be exhaustive, but rather to give an overview of frequently used notations.

Stochastic Hybrid Automata

- Var: Set of variables
- $\mathcal{V}$: Set of valuations (over a variable set)
- $v \in \mathcal{V}$: a valuation
- f: activity function specifying the evolution of variables in a location (do not confuse with $\tilde{f}$ in the context of the transformation)
- Con: Set of controlled variables
- $NCon$: Set of non-controlled variables. We always require that $Con \cap NCon = \emptyset$.
- $Distr$: Set of probability distributions
- δ_a: Dirac distribution assigning all probability mass to a
- $\mathcal{A} = (Loc, Var, Inv, Init, Edge, Act, Lab, Proc, Dur, Wgt)$: SHA
- Loc: Set of locations
- $loc \in Loc$: Location
- Inv: Assigns invariants to locations
- $Init$: Assigns initial valuations to locations
- $Edge$: Set of jumps
- $e \in Edge$: Jump
- Act: Assigns activities to locations
- Lab: Set of labels

- $a \in Lab$: Label
- $Proc$: Assigns labels to jumps
- Dur: Assigns probability distributions to jumps and valuations
- Wgt: Assigns a weight to each jump
- $\Theta^{\mathcal{A}}$: Set of states of $\mathcal{A}$
- $\vartheta = (loc, \nu, \nu_{Lab}) \in \Theta^{\mathcal{A}}$: State of $\mathcal{A}$
- ν_{Lab}: Valuation of the clock variables for every label
- $enabled_{\vartheta}(e)$: Holds iff jump e is enabled in ϑ
- $disc_{\vartheta}(e)$: Unique valuation reached from ϑ after taking jump e
- $fireable_{\vartheta}(e)$: Holds iff jump e is fireable in ϑ
- $time^{\tau}_{\vartheta}(f)$: Unique valuation reached from ϑ after letting τ time units pass and let the variables evolve according to f
- $clocks^{f}_{\vartheta}(\tau)$: Unique valuation of labels reached from ϑ after letting τ time units pass and let the variables evolve according to f
- $\vartheta \Rightarrow \vartheta'$: Holds iff ϑ' is reachable from ϑ by taking one or several semantic steps
- $\mathcal{A}_1 \parallel \mathcal{A}_2$: Parallel composition of $\mathcal{A}_1$ and $\mathcal{A}_2$

Hybrid Petri Nets with General Firings

- $\mathcal{H} = (\mathcal{P}, \mathcal{T}, \mathcal{A}, M_0, \Phi)$: HPnG
- $\mathcal{P}$: Set of places
- $\mathcal{P}^{disc}$: Set of discrete places
- $\mathcal{P}^{cont}$: Set of continuous places
- $p \in \mathcal{P}$: Place
- $\mathcal{T}$: Set of transitions
- $\mathcal{T}^{det}$: Set of deterministic transitions
- $\mathcal{T}^{gen}$: Set of general transitions
- $\mathcal{T}^{cont}$: Set of continuous transitions
- $\mathcal{T}^{disc}$: Set of discrete, i.e., deterministic or general transitions

- $t \in \mathcal{T}$: Transition
- $\mathcal{A}$: Set of arcs
- $\mathcal{A}^{disc}$: Set of discrete arcs
- $\mathcal{A}^{cont}$: Set of continuous arcs
- $\mathcal{A}^{test}$: Set of test arcs
- $\mathcal{A}^{inh}$: Set of inhibitor arcs
- $\langle p, t \rangle, \langle t, p \rangle \in \mathcal{A}$: Arc
- $M_0 = (m_0, x_0)$: Initial marking of discrete and continuous places
- Φ: Tuple of parameter functions
- $\Phi^{\mathcal{P}}_{ub}$: Upper bound for continuous places
- $\Phi^{\mathcal{T}}_{ft}$: Firing time for deterministic transitions
- $\Phi^{\mathcal{T}}_{gt}$: Probability distribution for general transitions
- $\Phi^{\mathcal{T}}_{fr}$: Nominal firing rate for continuous transitions
- $\Phi^{\mathcal{A}}_{w}$: Weight of non-continuous arcs
- $\Phi^{\mathcal{T}}_{p}$: Priority of discrete transitions
- $\mathcal{D}$: Discrete fragment of a HPnG
- $\mathcal{T}^{\mathcal{D}}$: Transitions included in discrete fragment $\mathcal{D}$
- $\mathcal{A}^{\mathcal{D}}$: Arcs included in discrete fragment $\mathcal{D}$
- $\Sigma^{\mathcal{H}}$: Set of states of $\mathcal{H}$
- $\sigma = (m, x, c, l) \in \Sigma^{\mathcal{H}}$: State of $\mathcal{H}$
- m: Marking of discrete places
- x: Marking of continuous places
- c: Clocks for discrete transitions
- l: Restriction list for continuous transitions and places
- $\mathcal{I}^{type}$: Input bag, i.e., input places of a transition or incoming transitions of a place connected via an arc of type *type*
- $\mathcal{O}^{type}$: Output bag, i.e., output places of a transition or outgoing transitions of a place connected via an arc of type *type*

- $conc_\sigma(t)$: Holds iff transition t has concession in σ
- $fireable_\sigma(t)$: Holds iff transition t is fireable in σ
- $rate_\sigma(t)$: Actual firing rate of continuous transition t in state σ
- $fire_t(m)$: Discrete marking after firing of discrete transition t
- $fire_t^\tau(m)$: Continuous marking after firing of continuous transition t for τ time units
- $evolve_T^\tau(c)$: Clocks updated for discrete transitions T after τ time units
- $reset_t(c)$: Clocks resampled for transition t
- $drift_\sigma(p)$: Drift of continuous place p in state σ
- $reduce_p^{empty}(l)$: Restriction list after rate reduction of continuous place p
- $restr_\sigma(p)$: Holds iff continuous place p restricts any transition in state σ
- $reset_p(l)$: Restriction list after rate reset of continuous place p
- $fireable_{\sigma,\tau}^{cont}$: Set of fireable continuous transitions in time span of duration τ starting in σ
- $conc_{\sigma,\tau}^{disc}$: Set of discrete transitions with concession in time span of duration τ starting in σ
- $safe_\sigma(\tau)$: Holds iff τ is a safe time span from σ
- $\sigma \Rightarrow \sigma'$: Holds iff σ' is reachable from σ by taking one or several semantic steps

Transforming HPnGs to SHAs

- $Var^{\mathcal{H}}$: Set of variables for a HPnG $\mathcal{H}$
- $\mathcal{V}^{\mathcal{H}}$: Set of valuations over $Var^{\mathcal{H}}$
- $conc_\nu(t)$: Holds iff transition t has concession in ν
- $fire_\nu(t)$: Valuation after firing of discrete transition t
- $\mathcal{A}_{\mathcal{D}}^{\mathcal{H}}$: SHA simulating the discrete fragment $\mathcal{D}$ of $\mathcal{H}$
- $rate_\nu(t)$: Actual firing rate of continuous transition t in valuation ν
- $drift_\nu(p)$: Drift of continuous place p in valuation ν
- $reduce_p^{empty}(\nu)$: Valuation (including restriction list) after rate reduction of continuous place p

- $restr_\nu(p)$: Holds iff continuous place p restricts any transition in valuation ν
- $reset_p(\nu)$: Valuation (including restriction list) after rate reset of continuous place p
- $\mathcal{A}_p^{\mathcal{H}}$: SHA simulating the continuous place p of $\mathcal{H}$
- $\mathcal{A}^{\mathcal{H}}$: SHA simulating the HPnG $\mathcal{H}$
- $@p$: Components originating from sub-SHA $\mathcal{A}_p^{\mathcal{H}}$ of place p
- $\tilde{f}$: Bisimulation between states of $\mathcal{H}$ and $\mathcal{A}^{\mathcal{H}}$
- $\tilde{f}^{-1}$: Inverse bisimulation between states of $\mathcal{H}$ and $\mathcal{A}^{\mathcal{H}}$

B. Omitted Proofs

B.1. Proof of Theorem 4.2

Let $\sigma \in \Sigma^{\mathcal{H}}$ and $\tilde{f}(\sigma) = (l, \nu, \nu_{Lab}) \in \Theta^{\mathcal{A}^{\mathcal{H}}}$.

Part 1. Let $t \in \mathcal{T}$. Assume that $conc_\sigma(t)$ holds. By Definition 3.6, we know that the following conditions hold:

1. $\forall p \in \mathcal{I}^{disc}(t) \colon m(p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle$
2. $\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{disc}) \colon m(p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle$
3. $\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{cont}) \colon x(p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle$
4. $\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{disc}) \colon m(p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle$
5. $\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{cont}) \colon x(p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle$

By definition of $\tilde{f}$, these conditions are equal to:

1. $\forall p \in \mathcal{I}^{disc}(t) \colon \nu(m_p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle$
2. $\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{disc}) \colon \nu(m_p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle$
3. $\forall p \in (\mathcal{I}^{inh}(t) \cap \mathcal{P}^{cont}) \colon \nu(x_p) < \Phi_w^{\mathcal{A}}\langle p, t\rangle$
4. $\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{disc}) \colon \nu(m_p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle$
5. $\forall p \in (\mathcal{I}^{test}(t) \cap \mathcal{P}^{cont}) \colon \nu(x_p) \geq \Phi_w^{\mathcal{A}}\langle p, t\rangle$

Therefore by Definition 4.2, $conc_\nu(t)$ holds and claim (1) is proven.

Part 2. For $t \in \mathcal{T}^{disc}$, we have:

$$
\begin{aligned}
& conc_\sigma(t) && \\
\iff\ & conc_\nu(t) && \text{by (1)} \\
\iff\ & (\nu, fire_\nu(t)) \in \mu \text{ where } jump_t = (loc, \mu, loc) && \text{by Definition 4.4} \\
\iff\ & enabled_{\tilde{f}(\sigma)}(jump_t) && \text{by Definition 2.7}
\end{aligned}
$$

Part 3. Let $t \in \mathcal{T}^{disc}$. Then, we have:

$$\begin{aligned}
& \mathit{fireable}_\sigma(t) \\
\iff\ & \mathit{conc}_\sigma(t) \wedge c(t) = 0 && \text{by Definition 3.7} \\
\iff\ & \mathit{enabled}_{\tilde{f}(\sigma)}(\mathit{jump}_t) \wedge c(t) = 0 && \text{by (2)} \\
\iff\ & \mathit{enabled}_{\tilde{f}(\sigma)}(\mathit{jump}_t) \wedge \nu_{Lab}(a_t) = 0 && \text{by def. of } \tilde{f} \\
\iff\ & \mathit{fireable}_{\tilde{f}(\sigma)}(\mathit{jump}_t) && \text{by Definition 2.9}
\end{aligned}$$

Part 4. Let $p \in \mathcal{P}^{cont}$. Then, we have:

$$\begin{aligned}
& \mathit{restr}_\sigma(p) \\
\iff\ & \exists t \in \mathcal{I}^{cont}(p) \cup \mathcal{O}^{cont}(p). \quad l(t)(p) < 1 && \text{by Definition 3.14} \\
\iff\ & \exists t \in \mathcal{I}^{cont}(p) \cup \mathcal{O}^{cont}(p). \quad l_{t,p} < 1 && \text{by def. of } \tilde{f} \\
\iff\ & \mathit{restr}_\nu(p) && \text{by Definition 4.8}
\end{aligned}$$

Part 5. Let $t \in \mathcal{T}^{cont}$. Then, we have:

$$\begin{aligned}
& \mathit{rate}_\sigma(t) \\
=\ & \begin{cases} 0 & \text{if } \neg \mathit{enabled}_\sigma(t) \\ \Phi^{\mathcal{T}}_{fr}(t) \cdot \min\left\{ l(t)(p) \mid p \in \mathcal{P}^{cont} \right\} & \text{if } \mathit{enabled}_\sigma(t) \end{cases} && \text{by Definition 3.8} \\
=\ & \begin{cases} 0 & \text{if } \neg \mathit{conc}_\sigma(t) \\ \Phi^{\mathcal{T}}_{fr}(t) \cdot \min\left\{ l(t)(p) \mid p \in \mathcal{P}^{cont} \right\} & \text{if } \mathit{conc}_\sigma(t) \end{cases} && \text{by Definition 3.7} \\
=\ & \begin{cases} 0 & \text{if } \neg \mathit{conc}_\nu(t) \\ \Phi^{\mathcal{T}}_{fr}(t) \cdot \min\left\{ l(t)(p) \mid p \in \mathcal{P}^{cont} \right\} & \text{if } \mathit{conc}_\nu(t) \end{cases} && \text{by (1)} \\
=\ & \begin{cases} 0 & \text{if } \neg \mathit{conc}_\nu(t) \\ \Phi^{\mathcal{T}}_{fr}(t) \cdot \min\left\{ \nu(l_{t,p}) \mid p \in \mathcal{P}^{cont} \right\} & \text{if } \mathit{conc}_\nu(t) \end{cases} && \text{by def. of } \tilde{f} \\
=\ & \mathit{rate}_\nu(t) && \text{by Definition 4.5}
\end{aligned}$$

Part 6. Let $p \in \mathcal{P}^{cont}$. Then, we have:

$$\begin{aligned}
& \mathit{drift}_\sigma(p) \\
=\ & \sum_{t_i \in \mathcal{I}^{cont}(p)} \mathit{rate}_\sigma(t_i) - \sum_{t_j \in \mathcal{O}^{cont}(p)} \mathit{rate}_\sigma(t_j) && \text{by Definition 3.12} \\
=\ & \sum_{t_i \in \mathcal{I}^{cont}(p)} \mathit{rate}_\nu(t_i) - \sum_{t_j \in \mathcal{O}^{cont}(P)} \mathit{rate}_\nu(t_j) && \text{by (5)} \\
=\ & \mathit{drift}_\nu(p) && \text{by Definition 4.6}
\end{aligned}$$

This concludes the proof of Theorem 4.2.

B.2. Proof of Theorem 4.1

We prove by induction that Theorem 4.1 holds using the bisimulation defined in Section 4.5. For the base case, we prove in Appendix B.2.1 that the set of initial states of $\mathcal{A}^{\mathcal{H}}$ is exactly the set of $\tilde{f}$ applied to all initial states of $\mathcal{H}$. In the induction step, we then show that in Appendix B.2.2 and the other direction in Appendix B.2.3.

B.2.1. Induction Base.

Part I. Let $\sigma \in \Sigma^{\mathcal{H}}$ be an initial state of $\mathcal{H}$. We are going to show that $\tilde{f}(\sigma) \in \Theta^{\mathcal{A}^{\mathcal{H}}}$ is an initial state of $\mathcal{A}^{\mathcal{H}}$. By Definition 3.4, we know that $\sigma = (m_0, x_0, c, \mathbf{1})$ where $c(t)$ is the firing time for deterministic transitions and sampled from the corresponding distribution for general transitions and $\mathbf{1}$ is the function mapping all entries of the restriction list to 1.

Then, $f(\sigma) = (loc, \nu, \nu_{Lab})$, where

- ν is the valuation where the markings correspond to the initial markings and the restriction entries equal 1, so

$$\nu(y) = \begin{cases} m_0(p) & \text{if } y = m_p \text{ for } p \in \mathcal{P}^{disc} \\ x_0(p) & \text{if } y = x_p \text{ for } p \in \mathcal{P}^{cont} \\ 1 & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont} \end{cases}$$

  for all $y \in Var^{\mathcal{H}}$.

- The values of the labels are corresponding to the sampled values for every discrete transition and 0 for all others, so

$$\nu_{Lab}(a) = \begin{cases} \Phi^{\mathcal{T}}_{ft}(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{gen} \\ 0 & \text{else} \end{cases}$$

  for all $a \in Lab$.

We argue that this is indeed an initial state of $\mathcal{A}^{\mathcal{H}}$ as defined in Definition 2.6. For this, we have to show that $\nu \in Init(loc)$ and argue that ν_{Lab} is a valid sample from the corresponding probability distributions. It is obvious that $\nu \in Init(loc)$, as it corresponds exactly to Definition 4.11.

For ν_{Lab}, we get that for all $a \in Lab$

$$\begin{aligned}
\nu_{Lab}(a) &= \begin{cases} \Phi^{\mathcal{T}}_{ft}(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{gen} \\ 0 & \text{else} \end{cases} \\
&= \begin{cases} \text{sample from } \delta_{\Phi^{\mathcal{T}}_{ft}(t)} & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{gen} \\ \text{sample from } \delta_0 & \text{else} \end{cases} && \text{by def. of } \delta \\
&= \text{sample from } Dur(a, (loc, \nu)) \text{ for all } \nu \in \mathcal{V}^{\mathcal{H}}
\end{aligned}$$

In conclusion, $\tilde{f}(\sigma)$ is an initial state of $\mathcal{A}^{\mathcal{H}}$.

Part II. Now, let $\vartheta \in \Theta^{\mathcal{A}^{\mathcal{H}}}$ be an initial state of $\mathcal{A}^{\mathcal{H}}$. We are going to show that $\tilde{f}^{-1}(\vartheta) \in \Sigma^{\mathcal{H}}$ is an initial state of $\mathcal{H}$. By Definition 2.6, we know that $\vartheta = (loc, \nu, \nu_{Lab})$ where $\nu \in Init(loc)$ and ν_{Lab} is sampled from the corresponding distribution.

Therefore, we know that $\nu(m_p) = m_0(p)$ for all $p \in \mathcal{P}^{disc}$, $\nu(x_p) = x_0(p)$ for all $p \in \mathcal{P}^{cont}$ and $\nu(l_{t,p}) = 1$ for all $p \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont}$. By definition of $\tilde{f}^{-1}$, we get that $\tilde{f}^{-1}(\vartheta) = (m_0, x_0, c, \mathbf{1})$, where c is given by

$$\begin{aligned}
c(t) &= \nu_{Lab}(a_t) && \text{by def. of } \tilde{f}^{-1} \\
&= \text{sample from } Dur(a_t, (loc, \nu)) && \text{by Definition 2.6} \\
&= \begin{cases} \text{sample from } \delta_{\Phi^{\mathcal{T}}_{ft}(t)} & \text{if } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } t \in \mathcal{T}^{gen} \end{cases} && \text{by Definition 4.11} \\
&= \begin{cases} \Phi^{\mathcal{T}}_{ft}(t) & \text{if } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } t \in \mathcal{T}^{gen} \end{cases} && \text{by def. of } \delta
\end{aligned}$$

for all $t \in \mathcal{T}^{disc}$. Therefore, $\tilde{f}^{-1}(\vartheta)$ is an initial state of $\mathcal{H}$ by Definition 3.4.

B.2.2. Induction Step Part I

We now show that for all $\sigma, \sigma' \in \Sigma^{\mathcal{H}}$ with $\sigma \Rightarrow \sigma'$, we have that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$. For this, we distinguish which semantic step is taken (see Definition 3.19).

Every subsection will be structured as follows: First, we discuss what we know about σ and σ' assuming that we reach σ' from σ applying a specific rule. Then, we examine what we can infer for $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$ based on the previous findings. Next, we present

the rule from the semantics of SHAs which will be used to simulate the step, and show that the corresponding premises are fulfilled. This lets us conclude that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$.

In this complete part of the proof, we let $\sigma = (m, x, c, l)$, $\sigma' = (m', x', c', l')$, $\tilde{f}(\sigma) = (loc, \nu, \nu_{Lab})$, and $\tilde{f}(\sigma') = (loc, \nu', \nu'_{Lab})$.

Firing of Deterministic Transitions

Assume that $\sigma \xRightarrow{\text{fire } t} \sigma'$, so the consequent state σ' is obtained by the firing of a deterministic transition $t \in \mathcal{T}^{disc}$. The semantic rule is given by

$$\frac{t \in \mathcal{T}^{disc} \quad fireable_{\sigma}(t)}{(m, x, c, l) \xRightarrow{\text{fire } t} (fire_t(m), x, reset_t(c), l)} \text{ fire}$$

We thus know that $\sigma' = (fire_t(m), x, reset_t(c), l)$ and that t is fireable in σ, i.e., $fireable_{\sigma}(t)$ is true.

We start by examining what we can infer about $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$.

- Obviously, we are in the same location loc in $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$.

- The only difference between the new valuation ν' and ν is that the discrete markings of the places connected to t potentially changed. Unsurprisingly, we have that $\nu' = fire_{\nu}(t)$, as we defined this with exactly the purpose of modeling

the firing of t. More formally, we have:

$$
\begin{aligned}
\nu'(y) &= \begin{cases} \mathit{fire}_t(m)(p) & \text{if } y = m_p \text{ for } p \in \mathcal{P}^{disc} \\ x(p) & \text{if } y = x_p \text{ for } p \in \mathcal{P}^{cont} \\ l(t)(p) & \text{if } y = l_{t,p} \text{ for } p \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont} \end{cases} && \text{by def. of } \tilde{f} \\
&= \begin{cases} \mathit{fire}_t(m)(p) & \text{if } y = m_p \text{ for } p \in \mathcal{P}^{disc} \\ \nu(y) & \text{else} \end{cases} \\
&= \begin{cases} m(p) - \Phi_w^{\mathcal{A}}\langle p, t\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{I}^{disc}(t) \\ m(p) + \Phi_w^{\mathcal{A}}\langle t, p\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{O}^{disc}(t) \\ m(p) & \text{if } y = m_p \text{ for all other } p \in \mathcal{P}^{disc} \\ \nu(y) & \text{else} \end{cases} && \text{by Definition 3.9} \\
&= \begin{cases} \nu(m_p) - \Phi_w^{\mathcal{A}}\langle p, t\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{I}^{disc}(t) \\ \nu(m_p) + \Phi_w^{\mathcal{A}}\langle t, p\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{O}^{disc}(t) \\ \nu(m_p) & \text{if } y = m_p \text{ for all other } p \in \mathcal{P}^{disc} \\ \nu(y) & \text{else} \end{cases} && \text{by def. of } \tilde{f} \\
&= \begin{cases} \nu(m_p) - \Phi_w^{\mathcal{A}}\langle p, t\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{I}^{disc}(t) \\ \nu(m_p) + \Phi_w^{\mathcal{A}}\langle t, p\rangle & \text{if } y = m_p \text{ for } p \in \mathcal{O}^{disc}(t) \\ \nu(y) & \text{else} \end{cases} \\
&= \mathit{fire}_\nu(t)(y) && \text{by Definition 4.3}
\end{aligned}
$$

- The new valuation of the labels ν'_{Lab} changed according to the reset of clocks. This means that the clock value for t is freshly sampled from the corresponding distribution. Let u be this new value. As defined in Definition 3.11, $u = \Phi_{ft}^{\mathcal{T}}(t)$ if t is a deterministic transition and u is sampled from $\Phi_{gt}^{\mathcal{T}}(t)$ if t is a general transition. Note that by definition of $\tilde{f}$, we have that $u \in supp(Dur(a_t, (loc, \nu)))$ as the probability distribution $Dur(a_t, (loc, \nu))$ either is $\delta_{\Phi_{ft}^{\mathcal{T}}(t)}$ or $\Phi_{gt}^{\mathcal{T}}(t)$ depending on the type of t.

Formally, we have that

$$\begin{aligned}
\nu'_{Lab}(a) &= \begin{cases} reset_t(c)(t') & \text{if } a = a_{t'} \text{ for } t' \in \mathcal{T}^{disc} \\ 0 & \text{else} \end{cases} && \text{by def. of } \tilde{f} \\
&= \begin{cases} c(t') & \text{if } a = a_{t'} \text{ for } t' \in \mathcal{T}^{disc} \text{ and } t' \neq t \\ \Phi^{\mathcal{T}}_{ft}(t) & \text{if } a = a_t \text{ and } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } a = a_t \text{ and } t \in \mathcal{T}^{gen} \\ 0 & \text{else} \end{cases} \\
& && \text{by Definition 3.11} \\
&= \begin{cases} c(t') & \text{if } a = a_{t'} \text{ for } t' \in \mathcal{T}^{disc} \text{ with } t' \neq t \\ u & \text{if } a = a_t \\ 0 & \text{else} \end{cases} && \text{by choice of } u \\
&= \nu_{Lab}[a_t \mapsto u](a)
\end{aligned}$$

for all $a \in Lab$. So, we know that $\nu'_{Lab} = \nu_{Lab}[a_t \mapsto u]$.

We aim to show that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$, and from the considerations above we know that

$$\tilde{f}(\sigma') = (loc, \nu', \nu'_{Lab}) = (loc, fire_\nu(t), \nu_{Lab}[a_t \mapsto u]).$$

In the transformation, the firing of a deterministic transition t is simulated by taking the respective jump $jump_t = (loc, \mu, loc)$ in the SHA of the corresponding discrete fragment. Therefore, we are going to show that

$$(loc, \nu, \nu_{Lab}) \xRightarrow{jump_t} (loc, fire_\nu(t), \nu_{Lab}[a_t \mapsto u]).$$

The rule as defined in Definition 2.12 is

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show for $jump_t$ that t is fireable in $\tilde{f}(\sigma)$, that the new valuation corresponds to the result of a discrete step, and that the new clock value is sampled correctly.

1. $fireable_{\tilde{f}(\sigma)}(jump_t)$ follows directly from $fireable_\sigma(t)$ and (3).
2. $disc_{\tilde{f}(\sigma)}(jump_t)$ is in Definition 2.8 defined as the unique valuation ν' reached after taking $jump_t = (loc, \mu, loc)$. By Definition 4.4, we have that

$$\mu = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; conc_\nu(t) \wedge \nu' = fire_\nu(t) \right\},$$

from which we can conclude that

$$disc_{\tilde{f}(\sigma)}(jump_t) = fire_\nu(t) = \nu'.$$

3. By Definition 4.4, we have that $Proc(jump_t) = a_t$. By choice of u, we also know that $u \in supp(Dur(a_t, (loc, \nu')))$. Therefore, the final premise also holds.

In conclusion, all premises hold and we have that

$$\sigma \xrightarrow{\text{fire } t} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \xrightarrow{jump_t} \tilde{f}(\sigma').$$

Rate Adaption: Reduce

Assume that $\sigma \xrightarrow{\text{reduce empty } p} \sigma'$ for a continuous place $p \in \mathcal{P}^{cont}$. The consequent state σ' is obtained by reducing the rates of the incoming transitions. The semantic rule is given by

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = 0 \quad drift_\sigma(p) < 0}{(m, x, c, l) \xrightarrow{\text{reduce empty } p} (m, x, c, reduce_p^{empty}(l))} \text{ reduce\_empty}$$

We thus know that p is empty in σ and has a negative drift. Additionally, we know that in σ', the list of restrictions l is modified to implement the new restrictions.

We start by examining what we can infer about $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$.

- The location of $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$ is loc.
- The new valuation ν' equals ν except that the list of restrictions is updated. More

formally, we have:

$$
\begin{aligned}
&\nu'(y)\\
&= \begin{cases} m(p') & \text{if } y = m_{p'} \text{ for } p' \in \mathcal{P}^{disc} \\ x(p') & \text{if } y = x_{p'} \text{ for } p' \in \mathcal{P}^{cont} \\ reduce_p^{empty}(l)(p')(t) & \text{if } y = l_{t,p} \text{ for } p \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont} \end{cases} && \text{by def. of } \tilde{f}\\
&= \begin{cases} reduce_p^{empty}(l)(t)(p') & \text{if } y = l_{t,p'} \text{ for } p' \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont} \\ \nu(y) & \text{else} \end{cases}\\
&= \begin{cases} \frac{in(p)}{out(p)} \cdot \min\left\{l(t)(p'') \mid p'' \in \mathcal{P}^{cont}\right\} & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{O}^{cont}(p).\, fireable_\sigma(t) \\ l(t)(p') & \text{if } y = l_{t,p'} \text{ for } t \in \mathcal{O}^{cont}(p), p' \in \mathcal{P}^{cont} \\ \nu(y) & \text{else} \end{cases}\\
& && \text{by Definition 3.13}\\
&= \begin{cases} \frac{in(p)}{out(p)} \cdot \min\left\{\nu(l_{t,p''}) \mid p'' \in \mathcal{P}^{cont}\right\} & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{O}^{cont}(p).\, fireable_\sigma(t) \\ \nu(y) & \text{else} \end{cases}\\
& && \text{by def. of } \tilde{f}\\
&= \begin{cases} \frac{in(p)}{out(p)} \cdot \min\left\{\nu(l_{t,p''}) \mid p'' \in \mathcal{P}^{cont}\right\} & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{O}^{cont}(p).\, conc_\sigma(t) \\ \nu(y) & \text{else} \end{cases}\\
& && \text{by Definition 3.7}\\
&= \begin{cases} \frac{in(p)}{out(p)} \cdot \min\left\{\nu(l_{t,p''}) \mid p'' \in \mathcal{P}^{cont}\right\} & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{O}^{cont}(p).\, conc_\nu(t) \\ \nu(y) & \text{else} \end{cases}\\
& && \text{by (1)}\\
&= reduce_p^{empty}(\nu)(y) && \text{by Definition 4.7}
\end{aligned}
$$

We can conclude that $\nu' = reduce_p^{empty}(\nu)$.

- Since the clock values do not change, by definition of $\tilde{f}$ we have that $\nu'_{Lab} = \nu_{Lab}$.

We aim to show that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$, and from the considerations above we know that

$$\tilde{f}(\sigma') = (loc, \nu', \nu'_{Lab}) = (loc, reduce_p^{empty}(\nu), \nu_{Lab}).$$

In the transformation, the rate adaption is simulated by several jumps. For the reduction, this is the jump $reduce_e@p$ from the sub-SHA corresponding to p.

$$(loc, \nu, \nu_{Lab}) \xRightarrow{reduce@p_e} (loc, reduce_p^{empty}(\nu), \nu_{Lab}).$$

The rule as defined in Definition 2.12 is

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that the jump is fireable in $\tilde{f}(\sigma)$, that the new valuation corresponds to the result of a discrete step, and that the new clock value is sampled correctly.

1. $fireable_{\tilde{f}(\sigma)}(reduce_e@p)$ is in Definition 2.9 defined to hold if $enabled_{\tilde{f}(\sigma)}(reduce_e@p)$ and $\nu_{Lab}(Proc(reduce_e@p)) = 0$. The jump $reduce_e@p$ is defined as (loc, μ_1, loc) with

$$\mu_1 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(x_p) = 0 \wedge drift_\nu(p) < 0 \wedge \nu' = reduce_p^{empty}(\nu) \right\}$$

   in Definition 4.10. By Definition 2.7, we know that $reduce_e@p$ is enabled if $(\nu, \nu'') \in \mu_1$ for some $\nu'' \in \mathcal{V}$. Therefore, we need to show that $\nu(x_p) = 0$ and that $drift_\nu(p) < 0$. We know that $x(p) = 0$, which implies the former by definition of $\tilde{f}$. We also know that $drift_\sigma(p) < 0$, and with (6) this implies $drift_\nu(p) < 0$. Therefore, ν fulfills the guard of the jump $reduce_e@p$ and with that, $enabled_{\tilde{f}(\sigma)}(reduce_e@p)$ holds.

   In Definition 4.10 we defined $Proc(reduce_e@p) = a_1@p$. The assigned distribution is δ_0, by which we know that $a_1@p$ is never assigned a value other than zero. Consequently, $\nu_{Lab}(Proc(reduce_e@p)) = 0$ is true. In summary, $fireable_{\tilde{f}(\sigma)}(reduce_e@p)$ holds.

2. Since $(\nu, \nu') = (\nu, reduce_p^{empty}(\nu)) \in \mu_1$ by above considerations, we know that $\nu' = disc_{\tilde{f}(\sigma)}(reduce_e@p)$.

3. As argued above, the label $a_1@p$ is never assigned a value other than zero. In particular, $u \in supp(Dur(a_1@p, (loc, \nu'))) = supp(\delta_0) = \{0\}$ implies that $u = 0$ and with that we obtain $\nu_{Lab}[a_1@p \mapsto 0] = \nu_{Lab}$.

In conclusion, all premises hold and we have that

$$\sigma \xRightarrow{\text{reduce empty } p} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \xRightarrow{reduce@p_e} \tilde{f}(\sigma').$$

The dual statement can be shown for p at the upper boundary, i.e., we have that

$$\sigma \xRightarrow{\text{reduce full } p} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \xRightarrow{reduce@p_f} \tilde{f}(\sigma').$$

Rate Adaption: Reset

Assume that $\sigma \xRightarrow{\text{reset empty } p} \sigma'$ for a continuous place $p \in \mathcal{P}^{cont}$. The consequent state σ' is obtained by resetting the restrictions on the incoming transitions. The semantic

rule is given by

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = 0 \quad drift_\sigma(p) > 0 \quad restr_\sigma(p)}{(m, x, c, l) \xrightarrow{\text{reset empty } p} (m, x, c, reset_p(l))} \text{ reset\_empty}$$

We thus know that in σ, p is empty, places restrictions and has a positive drift. Additionally, we know that in σ', the list of restrictions l is modified to implement the new restrictions.

We start by examining what we can infer about $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$.

- The location of $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$ is loc.
- The new valuation ν' equals ν except for the list of restrictions. More formally, we have:

$$\begin{aligned}
\nu'(y) &= \begin{cases} m'(p') & \text{if } y = m_{p'} \text{ for } p' \in \mathcal{P}^{disc} \\ x'(p') & \text{if } y = x_{p'} \text{ for } p' \in \mathcal{P}^{cont} \\ reset_p(l)(p')(t) & \text{if } y = l_{t,p'} \text{ for } p' \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont} \end{cases} && \text{by def. of } \tilde{f} \\
&= \begin{cases} reset_p(l)(t)(p') & \text{if } y = l_{t,p'} \text{ for } p' \in \mathcal{P}^{cont}, t \in \mathcal{T}^{cont} \\ \nu(y) & \text{else} \end{cases} \\
&= \begin{cases} 1 & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{T}^{cont} \\ l(t)(p') & \text{if } y = l_{t,p'} \text{ for } p' \in \mathcal{P}^{cont} \backslash \{p\}, t \in \mathcal{T}^{cont} \\ \nu(y) & \text{else} \end{cases} && \text{by Definition 3.15} \\
&= \begin{cases} 1 & \text{if } y = l_{t,p} \text{ for } t \in \mathcal{T}^{cont} \\ \nu(y) & \text{else} \end{cases} \\
&= reset_p(\nu)(y) && \text{by Definition 4.9}
\end{aligned}$$

  Therefore, we know that $\nu' = reset_p(\nu)$.

- Since the clock values do not change, by definition of $\tilde{f}$ we have that $\nu'_{Lab} = \nu_{Lab}$.

We aim to show that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$, and from the considerations above we know that

$$\tilde{f}(\sigma') = (loc, \nu', \nu'_{Lab}) = (loc, reset_p(\nu), \nu_{Lab}).$$

In the transformation, this step of the rate adaption is again simulated by a jump, in this case $reset_e@p$. Therefore, we are going to show that

$$(loc, \nu, \nu_{Lab}) \xrightarrow{reset@p_e} (loc, reset_p(\nu), \nu_{Lab}).$$

The rule as defined in Definition 2.12 is again

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that the jump is fireable in $\tilde{f}(\sigma)$, that the new valuation corresponds to the result of a discrete step, and that the new clock value is sampled correctly.

1. $fireable_{\tilde{f}(\sigma)}(reset_e@p)$ is in Definition 2.9 defined to hold if $enabled_{\tilde{f}(\sigma)}(reset_e@p)$ and $\nu_{Lab}(Proc(reset_e@p)) = 0$. The jump $reset_e@p$ is defined as (loc, μ_3, loc) with

   $$\mu_3 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \nu(x_p) = 0 \wedge restr_\nu(p) \wedge drift_\nu(p) > 0 \wedge \nu' = reset_p(\nu) \right\}$$

   in Definition 4.10. We know by assumption that $x(p) = 0$, $drift_\sigma(p) > 0$ and $restr_\sigma(p)$ holds. By definition of $\tilde{f}$ along with (6) and (4), we can immediately follow that ν satisfies the guard of the jump $reset_e@p$ and with that, $enabled_{\tilde{f}(\sigma)}(reset_e@p)$ holds.

   Additionally, in Definition 4.10 we defined $Proc(reset_e@p) = a_3@p$. The assigned distribution is δ_0, by which we know that $a_3@p$ is never assigned a value other than zero. Consequently, $\nu_{Lab}(Proc(reset_e@p)) = 0$ is true. In summary, $fireable_{\tilde{f}(\sigma)}(reset_e@p)$ holds.

2. Since $(\nu, \nu') = (\nu, reset_p(\nu)) \in \mu_3$ by above considerations, we know that $\nu' = disc_{\tilde{f}(\sigma)}(reset_e@p)$.

3. As argued above, the label $a_3@p$ is never assigned a value other than zero. In particular, $u \in supp(Dur(a_3@p, (loc, \nu'))) = supp(\delta_0)$ implies that $u = 0$ and with that we obtain $\nu_{Lab}[a_3@p \mapsto 0] = \nu_{Lab}$.

In conclusion, all premises hold and we have that

$$\sigma \xRightarrow{\text{reset empty } p} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \xRightarrow{reset@p_e} \tilde{f}(\sigma').$$

In the dual case for p at upper boundary, we have that

$$\sigma \xRightarrow{\text{reset full } p} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \xRightarrow{reset@p_f} \tilde{f}(\sigma').$$

Rate Adaption: Reset Single

Assume that $\sigma \xRightarrow{\text{reset } t, p} \sigma'$ for $t \in \mathcal{T}^{cont}$ and $p \in \mathcal{P}^{cont}$. The consequent state σ' is obtained by resetting the restriction list entry of t and p to one. The semantic rule is

given by

$$\frac{t \in \mathcal{T}^{cont} \quad \neg fireable_\sigma(t) \quad p \in \mathcal{P}^{cont} \quad l(t)(p) < 1}{(m,x,c,l) \xRightarrow{\text{reset } t,p} (m,x,c,l[t,p \mapsto 1])} \text{ reset\_single}$$

We thus know that in σ, t is not fireable and p places restrictions on t, and that the restriction list in σ' is updated to one for t and p.

We start by examining what we can infer about $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$.

- The location of $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$ is loc.
- The new valuation ν' equals ν except that the restriction list at index t,p is set to one. More formally, we have:

$$\begin{aligned}
\nu'(y) &= \begin{cases} m'(p') & \text{if } y = m_{p'} \text{ for } p' \in \mathcal{P}^{disc} \\ x'(p') & \text{if } y = x_{p'} \text{ for } p' \in \mathcal{P}^{cont} \\ l[t,p \mapsto 1](p')(t') & \text{if } y = l_{t',p'} \text{ for } p' \in \mathcal{P}^{cont}, t' \in \mathcal{T}^{cont} \end{cases} && \text{by def. of } \tilde{f} \\
&= \begin{cases} 1 & \text{if } y = l_{t,p} \\ \nu(y) & \text{else} \end{cases} \\
&= \nu[l_{t,p} \mapsto 1]
\end{aligned}$$

- Since the clock values do not change, by definition of $\tilde{f}$ we have that $\nu'_{Lab} = \nu_{Lab}$.

We aim to show that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$, and from the considerations above we know that

$$\tilde{f}(\sigma') = (loc, \nu', \nu'_{Lab}) = (loc, \nu[l_{t,p} \mapsto 1], \nu_{Lab}).$$

In the transformation, the maintenance of restriction lists is simulated by the jump *resetSingle@p* in the SHA corresponding to p. Therefore, we are going to show that

$$(loc, \nu, \nu_{Lab}) \xRightarrow{resetSingle@p} (loc, \nu[l_{t,p} \mapsto 1], \nu_{Lab}).$$

The rule as defined in Definition 2.12 is

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that the jump is fireable in $\tilde{f}(\sigma)$, that the new valuation corresponds to the result of a discrete step, and that the new clock value is sampled correctly.

1. As defined in Definition 2.9, $\mathit{fireable}_{\tilde{f}(\sigma)}(\mathit{resetSingle@p})$ holds if $\mathit{resetSingle@p}$ is enabled, i.e., $\mathit{enabled}_{\tilde{f}(\sigma)}(\mathit{resetSingle@p})$ holds, and the clock has run out, i.e., $\nu_{Lab}(\mathit{Proc}(\mathit{resetSingle@p})) = 0$. The jump $\mathit{resetSingle@p}$ is defined as $(\mathit{loc}, \mu_5, \mathit{loc})$ with

   $$\mu_5 = \left\{ (\nu, \nu') \in \mathcal{V}^{\mathcal{H}} \times \mathcal{V}^{\mathcal{H}} \;\middle|\; \exists t \in \mathcal{T}^{cont}.\ \neg \mathit{conc}_{\nu}(t) \wedge \nu(l_{t,p}) < 1 \wedge \nu' = \nu[l_{t,p} \mapsto 1] \right\}$$

   in Definition 4.10. We know by assumption that $\neg \mathit{fireable}_{\sigma}(t)$ and $l(t)(p) < 1$. Since t is a continuous transition, $\neg \mathit{fireable}_{\sigma}(t)$ implies that $\neg \mathit{conc}_{\sigma}(t)$ by Definition 3.7. Then, with (1) and the definition of $\tilde{f}$, we can conclude that ν satisfies the guard of the jump $\mathit{resetSingle@p}$ and with that, $\mathit{enabled}_{\tilde{f}(\sigma)}(\mathit{resetSingle@p})$ holds.

   Additionally, in Definition 4.10 we defined $\mathit{Proc}(\mathit{resetSingle@p}) = a_5@p$. The assigned distribution is δ_0, by which we know that $a_5@p$ is never assigned a value other than zero. Consequently, $\nu_{Lab}(\mathit{Proc}(\mathit{resetSingle@p})) = 0$ is true. In summary, $\mathit{fireable}_{\tilde{f}(\sigma)}(\mathit{resetSingle@p})$ holds.

2. Since $(\nu, \nu') = (\nu, \nu[l_{t,p} \mapsto 1]) \in \mu_5$ by above considerations, we know that $\nu' = \mathit{disc}_{\tilde{f}(\sigma)}(\mathit{resetSingle@p})$.

3. As argued above, the label $a_5@p$ is never assigned a value other than zero. In particular, $u \in \mathit{supp}(\mathit{Dur}(a_5@p, (\mathit{loc}, \nu'))) = \mathit{supp}(\delta_0)$ implies that $u = 0$ and with that we obtain $\nu_{Lab}[a_5@p \mapsto 0] = \nu_{Lab}$.

In conclusion, all premises hold and we have that

$$\sigma \xRightarrow{\text{reset } t, p} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \xRightarrow{\mathit{resetSingle@p}} \tilde{f}(\sigma').$$

Passing of Time

Assume that $\sigma \xRightarrow{\tau} \sigma'$, so we are considering the passing of $\tau \in \mathbb{R}_{>0}$ time units. The corresponding semantic rule is given by

$$\frac{\tau \in \mathbb{R}_{>0} \quad \mathit{safe}_{\sigma}(\tau)}{(m, x, c, l) \xRightarrow{\tau} (m, \mathit{fire}^{\tau}_{\mathit{fireable}^{cont}_{\sigma,\tau}}(x), \mathit{evolve}^{\tau}_{\mathit{conc}^{disc}_{\sigma,\tau}}(c), l)} \text{ time}$$

We thus know that τ is a safe time span from σ on as defined in Definition 3.18 and that

$$\sigma' = (m, \mathit{fire}^{\tau}_{T_1}(x), \mathit{evolve}^{\tau}_{T_2}(c), l)$$

with $T_1 = \mathit{fireable}^{cont}_{\sigma,\tau}$ and $T_2 = \mathit{conc}^{disc}_{\sigma,\tau}$ being the relevant sets of continuous respectively discrete transitions.

We start by examining what we can infer about $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$.

- The location of $\tilde{f}(\sigma)$ and $\tilde{f}(\sigma')$ is loc.
- In the new valuation ν', only the continuous markings are changed. We defined $fire^{\tau}_{T_1}(x)$ in Definition 3.9 as

$$fire^{\tau}_{T_1}(x)(p) = x(p) + \tau \cdot \sum_{t \in \mathcal{I}^{cont}(p) \cap T_1} rate_{\sigma}(t) - \tau \cdot \sum_{t \in \mathcal{O}^{cont}(p) \cap T_1} rate_{\sigma}(t)$$

for each continuous place p. Next, recall that T_1 only contains fireable continuous transitions by Definition 3.16. Therefore, we know that the above equality can be simplified to

$$fire^{\tau}_{T_1}(x)(p) = x(p) + \tau \cdot drift_{\sigma}(p)$$

for each continuous place p by Definition 3.12.

This exactly corresponds to the activity function given for every p in the corresponding SHA, and with that in the transformation. We defined the set of activity functions as those that are solutions of

$$\dot{x}_p = drift_{\nu}(p).$$

In particular, we have that there exists a unique $\hat{f} \in Act(loc)$ with

$$\hat{f}(0)(y) = \nu(y)$$

and

$$\hat{f}(\tau)(y) = \begin{cases} \nu(x_p) + \tau \cdot drift_{\nu}(p) & \text{if } x_p \text{ for } p \in \mathcal{P}^{cont} \\ \nu(y) & \text{else.} \end{cases}$$

These observations in combination with the fact that $drift_{\sigma}(p) = drift_{\nu}(p)$ by (6) and $x(p) = \nu(x_p)$ by construction lets us conclude that $\nu = \hat{f}(0)$ and that $\nu' = \hat{f}(\tau)$.

- The clock values in the new state are given by $evolve^{\tau}_{T_2}(c)$, which intuitively decreases the clock for every transition in T_2 by τ time units. The same is reflected in ν'_{Lab}. Formally, we have that

$$\begin{aligned} \nu'_{Lab}(a) &= \begin{cases} evolve^{\tau}_{T_2}(c)(t) & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{disc} \\ 0 & \text{else} \end{cases} && \text{by def. of } \tilde{f} \\ &= \begin{cases} c(t) - \tau & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{disc} \cap T_2 \\ c(t) & \text{if if } a = a_t \text{ for } t \in \mathcal{T}^{disc} \backslash T_2 \\ 0 & \text{else} \end{cases} && \text{by Definition 3.10} \\ &= \begin{cases} c(t) - \tau & \text{if } a = a_t \text{ for } t \in \mathcal{T}^{disc} \cap T_2 \\ \nu_{Lab}(t) & \text{else} \end{cases} && \text{by def. of } \tilde{f} \end{aligned}$$

for all $a \in Lab$. Thus, we know that ν'_{Lab} updates exactly the labels corresponding to a jump from T_2. We denote this by $\nu'_{Lab} = \nu_{Lab}[T_2 \multimap\!= \tau]$.

We aim to show that $\tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma')$, and from the considerations above we know that

$$\tilde{f}(\sigma') = (loc, \nu', \nu'_{Lab}) = (loc, \hat{f}(\tau), \nu_{Lab}[T_2 \multimap\!= \tau]).$$

As expected, we simulate the passing of time in the HPnGby letting time pass in the transformation. Therefore, we are going to show that

$$(loc, \nu, \nu_{Lab}) \xRightarrow{\tau, \hat{f}} (loc, \hat{f}(\tau), \nu_{Lab}[T_2 \multimap\!= \tau]).$$

The corresponding rule defined in Definition 2.12 is

$$\frac{f \in Act(l) \quad \tau \in \mathbb{R}_{>0} \quad \nu' = time^{f}_{\vartheta}(\tau) \quad \nu'_{Lab} = clocks^{f}_{\vartheta}(\tau)}{(l, \nu, \nu_{Lab}) \xRightarrow{\tau, f} (l, \nu', \nu'_{Lab})} \text{ time}$$

With the above observations in mind, we now prove that this rule is indeed applicable. Concretely, this means that we have to show that ν' is reached from ν after τ time steps and that the clocks are updated correctly.

1. $\nu' = time^{\hat{f}}_{\tilde{f}(\sigma)}(\tau)$ is defined in Definition 2.10 to be true if

   - $\hat{f}(0) = \nu$,
   - $\hat{f}(\tau) = \nu'$, and
   - $\hat{f}(\tau') \in Inv(loc)$ for all $\tau' \in [0, \tau]$.

   The correctness of the first two points were already proven above, abd the third point holds trivially since Inv allows all valuations. In conclusion, we have that $\nu' = time^{\hat{f}}_{\tilde{f}(\sigma)}(\tau)$.

2. $clocks^{\hat{f}}_{\sigma}(\tau) = \nu'_{Lab}$ is true if all jumps corresponding to a label are either enabled in the complete time interval or not at all, and that the clocks are decreased accordingly. For formal details, see Definition 2.11.

   To show that this holds, we split the set of labels into those that belong to a transition in $t \in T_2$ and those that do not. We start with the former and let $A = \{a_t \mid t \in T_2\} \subseteq Lab$ be those labels.

   Then, for every $a_t \in A$, there exists a jump $jump_t \in Edge$ with $Proc(jump_t) = a_t$. We have to show that

   - for all $\tau' \in (0, \tau)$ there exists $\nu'' \in \mathcal{V}$ such that $\nu'' = time^{\hat{f}}_{\tilde{f}(\sigma)}(\tau')$ and $enabled_{\vartheta'}(jump_t)$ for $\vartheta' = (l, \nu'', \nu_{Lab})$, and

- $\nu'_{Lab}(a_t) = \nu_{Lab}(a_t) - \tau \geq 0$.

The second part was already shown above. The first part basically means that $jump_t$ is enabled after τ' time units for $\tau' \in (0, \tau)$. To prove this, first note that $\nu'' = time^{\hat{f}}_{\tilde{f}(\sigma)}(\tau') = \hat{f}(\tau')$ is implied by the first part of this proof, since $\tau' \in [0, \tau]$. Therefore, the existence of ν'' is guaranteed.

Next, we argue that $enabled_{\vartheta'}(jump_t)$ holds, which is the case if $conc_{\nu''}(jump_t)$ holds by definition of $jump_t$. From the fact that τ is a safe time span, we know that $enabled_{\sigma}(t) = enabled_{\sigma''}(t)$ for all intermediate states σ''. Together with (2), this implies that t has concession in the valuation corresponding to σ'. In particular, $conc_{\nu''}(jump_t)$ holds. Therefore, the claim holds for all labels $a \in A$.

Next, we consider the labels $a \notin A$, i.e., those that are not associated with a transition in T_2. We need to argue that all jumps corresponding to those labels do not become enabled in any intermediate state. More formally, we have to show that for every jump e corresponding to a label $a \notin A$, it holds that

- for all $\tau' \in (0, \tau)$, there is no valuation $\nu'' \in \mathcal{V}$ with $\nu'' = time^{f}_{\vartheta}(\tau')$ and $enabled_{\vartheta'}(e)$ for $\vartheta' = (l, \nu'', \nu_{Lab})$, and
- $\nu'_{Lab}(a) = \nu_{Lab}(a)$.

Again, the second property was already shown above. As in the previous part, we know that $\nu'' = time^{\hat{f}}_{\tilde{f}(\sigma)}(\tau') = \hat{f}(\tau')$. It remains to prove that $enabled_{\vartheta'}(e)$ does not hold in any intermediate state. To do so, we distinguish between the concrete types of labels:

- The first labels we consider are of the form a_t for jumps $jump_t$ with $t \in \mathcal{T}^{disc} \setminus T_2$, which are enabled if t has concession. We know that there is an intermediate state in which t does not have concession, otherwise it would be included in T_2 by Definition 3.17. By Definition 3.18, we know that the concession status does not change within the interval, i.e., t does not have concession in any intermediate state. This implies with (2) that $jump_t$ is not enabled in any intermediate valuation, so $enabled_{\vartheta'}(jump_t)$ does not hold.
- For every continuous place $p \in \mathcal{P}^{cont}$, we have a label $a_1@p$ corresponding to the jump $reduce_e@p$ originating from the sub-SHA corresponding to p. These jumps are by Definition 4.10 enabled for a valuation ν'' if $\nu''(x_p) = 0$ and $drift_{\nu''}(p) < 0$. Since τ is a safe time span, we know that $x''(p) = 0$ implies that $drift_{\sigma''}(p) \geq 0$ for all intermediate states σ''. Again with (6) and the definition of $\tilde{f}$, this implies that $\nu''(x_p) = 0 \wedge drift_{\nu''}(p) < 0$ never holds. Therefore, $enabled_{\vartheta'}(reduce_e@p)$ does not hold.
- For every continuous place $p \in \mathcal{P}^{cont}$, we have a label $a_2@p$ corresponding to

the jump $reduce_f@p$ originating from the sub-SHA corresponding to p. With dual argumentation as above, we have that $enabled_{\vartheta'}(reduce_f@p)$ does not hold.

- For every continuous place $p \in \mathcal{P}^{cont}$, we have a label $a_3@p$ corresponding to the jump $reset_e@p$ originating from the sub-SHA corresponding to p. Such a jump is by Definition 4.10 enabled for a valuation ν'' if $\nu''(x_p) = 0$, $restr_{\nu''}(p)$, and $drift_{\nu''}(p) > 0$. Since τ is a safe time span, we know that $x''(p) = 0$ implies that for all intermediate states σ'', we either have $drift_{\sigma''}(p) = 0$ or $drift_{\sigma''}(p) > 0$ and $\neg restr_{\sigma''}(p)$. By (4) and (6), it follows that the required properties for $reset_e@p$ cannot hold at the same time, thus $enabled_{\vartheta'}(reset_e@p)$ does not hold.

- For every continuous place $p \in \mathcal{P}^{cont}$, we have a label $a_4@p$ corresponding to the jump $reset_f@p$ originating from the sub-SHA corresponding to p. With dual argumentation as above, we have that $enabled_{\vartheta'}(reduce_f@p)$ does not hold.

- For every continuous place $p \in \mathcal{P}^{cont}$, we have a label $a_5@p$ corresponding to the jump $resetSingle@p$ originating from the sub-SHA corresponding to p. These jumps are by Definition 4.10 enabled for a valuation ν'' if there exists a continuous transition $t \in \mathcal{T}^{cont}$ that does not have concession but is restricted, i.e, $\neg conc_{\nu''}(t)$ and $\nu''(l_{t,p}) < 1$. Since τ is a safe time span, we know that $\neg fireable_{\sigma''}(t)$ implies $l(t)(p) = 1$ for all intermediate states σ''. For continuous transitions, the notions of being fireable and having concession coincide, therefore we also know that $\neg conc_{\sigma''}(t)$. By (1) and the definition of $\tilde{f}$, it follows that $\neg conc_{\nu''}(t)$ implies $\nu''(l_{t,p}) = 1$, i.e., there does not exist a transition that would be required for $resetSingle@p$ to fire. In conclusion, $enabled_{\vartheta'}(resetSingle@p)$ does not hold.

We can conclude that $clocks_\sigma^{\hat{f}}(\tau) = \nu'_{Lab}$.

In conclusion, all premises hold and we have that

$$\sigma \overset{\tau}{\Longrightarrow} \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \overset{\tau,\hat{f}}{\Longrightarrow} \tilde{f}(\sigma').$$

Conclusion. We now have considered every possible semantic rule for HPnGs. Therefore, we have proven that

$$\sigma \Rightarrow \sigma' \quad \text{implies} \quad \tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma').$$

The other direction is shown in the following section.

B.2.3. Induction Step Part II

We now show that for $\vartheta, \vartheta' \in \Theta^{\mathcal{A}^{\mathcal{H}}}$ with $\vartheta \Rightarrow \vartheta'$, we have $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$.

Again, we structure the subsections as follows: First, we discuss what we know about ϑ and ϑ' assuming that we reach ϑ' from ϑ applying a specific rule. The semantics of SHAs are described in three rules: one for taking jumps, where we will additionally distinguish between which jump is taken, one for the passing of time, and one for changes made by the environment. Since we do not have any non-controlled variables, we do not have to consider the environment rule. Then, we examine what we can infer for $\tilde{f}^{-1}(\vartheta)$ and $\tilde{f}^{-1}(\vartheta')$ based on the previous findings. Next, we present the rule from the semantics of HPnGs which will be used to simulate the step, and show that the corresponding premises are fulfilled. This lets us conclude that $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$.

We denote $\vartheta = (l, \nu, \nu_{Lab})$, $\vartheta' = (l', \nu', \nu'_{Lab})$, $\tilde{f}^{-1}(\vartheta) = \sigma = (m, x, c, l)$, and $\tilde{f}^{-1}(\vartheta') = \sigma' = (m', x', c', l')$.

Firing of Deterministic Transitions

Assume that $\vartheta \xRightarrow{\text{jump}_t} \vartheta'$ for some discrete transition $t \in \mathcal{T}^{disc}$, so the consequent state ϑ' is obtained by taking a jump corresponding to a discrete transition. The semantic rule is given in Definition 2.12 by

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

We thus know that $fireable_\vartheta(jump_t)$, $\nu' = disc_\vartheta(jump_t)$, and $\nu'_{Lab} = \nu_{Lab}[Proc(e) \mapsto u]$ for an $u \in supp(Dur(a_t, (loc, \nu')))$.

We start by examining what we can infer about $\tilde{f}^{-1}(\vartheta)$ and $\tilde{f}^{-1}(\vartheta')$.

- For every discrete place $p \in \mathcal{P}^{disc}$, it holds that

$$\begin{aligned}
m'(p) &= \nu'(m_p) && \text{by def. of } \tilde{f}^{-1} \\
&= fire_\nu(t)(m_p) && \text{since } \nu' = disc_\vartheta(jump_t) \\
&= \begin{cases} \nu(m_p) - \Phi^{\mathcal{A}}_w\langle p, t\rangle & \text{if } p \in \mathcal{I}^{disc}(t) \\ \nu(m_p) + \Phi^{\mathcal{A}}_w\langle t, p\rangle & \text{if } p \in \mathcal{O}^{disc}(t) \\ \nu(m_p) & \text{else} \end{cases} && \text{by Definition 4.3} \\
&= \begin{cases} m(p) - \Phi^{\mathcal{A}}_w\langle p, t\rangle & \text{if } p \in \mathcal{I}^{disc}(t) \\ m(p) + \Phi^{\mathcal{A}}_w\langle t, p\rangle & \text{if } p \in \mathcal{O}^{disc}(t) \\ m(p) & \text{else} \end{cases} && \text{by def. of } \tilde{f}^{-1} \\
&= fire_t(m)(p) && \text{by Definition 3.9}
\end{aligned}$$

So, we have that $m' = \mathit{fire}_t(m)$.

- For every continuous place $p \in \mathcal{P}^{cont}$, it holds that

$$\begin{aligned} x'(p) &= \nu'(x_p) && \text{by def. of } \tilde{f}^{-1} \\ &= \mathit{fire}_\nu(t)(x_p) && \text{since } \nu' = \mathit{disc}_\vartheta(\mathit{jump}_t) \\ &= \nu(x_p) && \text{by Definition 4.3} \\ &= x(p) && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

So, we have that $x' = x$.

- For every discrete transition $t' \in \mathcal{T}^{disc}$, we have that

$$\begin{aligned} c'(t') &= \nu'_{Lab}(a_{t'}) && \text{by def. of } \tilde{f}^{-1} \\ &= \nu_{Lab}[a_t \mapsto u](a_{t'}) \\ &= \begin{cases} \nu_{Lab}(a_{t'}) & \text{if } t' \neq t \\ u \in \mathit{supp}(\mathit{Dur}(a_t, (l', v'))) & \text{if } t' = t \end{cases} \\ &= \begin{cases} \nu_{Lab}(a_{t'}) & \text{if } t' \neq t \\ u \in \mathit{supp}(\delta_{\Phi^{\mathcal{T}}_{ft}(t)}) & \text{if } t' = t \text{ and } t \in \mathcal{T}^{det} \\ u \in \mathit{supp}(\Phi^{\mathcal{T}}_{gt}(t)) & \text{if } t' = t \text{ and } t \in \mathcal{T}^{gen} \end{cases} && \text{by Definition 4.4} \\ &= \begin{cases} \nu_{Lab}(a_{t'}) & \text{if } t' \neq t \\ \Phi^{\mathcal{T}}_{ft}(t) & \text{if } t' = t \text{ and } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } t' = t \text{ and } t \in \mathcal{T}^{gen} \end{cases} \\ &= \begin{cases} c(t') & \text{if } t' \neq t \\ \Phi^{\mathcal{T}}_{ft}(t) & \text{if } t \in \mathcal{T}^{det} \\ \text{sample from } \Phi^{\mathcal{T}}_{gt}(t) & \text{if } t \in \mathcal{T}^{gen} \end{cases} && \text{by def. of } \tilde{f}^{-1} \\ &= \mathit{reset}_t(c)(t') && \text{by Definition 3.11} \end{aligned}$$

So, we have that $c' = \mathit{reset}_t(c)$.

- For every continuous transition $t' \in \mathcal{T}^{cont}$ and place $p \in \mathcal{P}^{cont}$, it holds that

$$\begin{aligned} l'(t')(p) &= \nu'(l_{t',p}) && \text{by def. of } \tilde{f}^{-1} \\ &= \mathit{fire}_\nu(t)(l_{t',p}) && \text{since } \nu' = \mathit{disc}_\vartheta(\mathit{jump}_t) \\ &= \nu(l_{t',p}) && \text{by Definition 4.3} \\ &= l(t')(p) && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

So, we have that $l' = l$.

We aim to show that $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$, and from the considerations above we know that

$$\tilde{f}^{-1}(\vartheta') = (m', x', c', l') = (m, fire_t(m), reset_t(c), l).$$

In the transformation, taking $jump_t$ simulates the firing of transition t. Therefore, we are going to show that

$$(m, x, c, l) \xRightarrow{\text{fire } t} (m, fire_t(m), reset_t(c), l).$$

The rule as defined in Section 3.2.3 is

$$\frac{t \in \mathcal{T}^{disc} \quad fireable_\sigma(t)}{(m, x, c, l) \xRightarrow{\text{fire } t} (fire_t(m), x, reset_t(c), l)} \text{ fire}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that $fireable_{\tilde{f}^{-1}(\vartheta)}(t)$. We know by assumption that $fireable_\vartheta(jump_t)$. By (3), this directly implies that $fireable_{\tilde{f}^{-1}(\vartheta)}(t)$.

In conclusion, all premises hold and we have that

$$\vartheta \xRightarrow{\text{jump}_t} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) \xRightarrow{\text{fire } t} \tilde{f}^{-1}(\vartheta').$$

Rate Adaption: Reduce

Assume that $\vartheta \xRightarrow{reduce_e@p} \vartheta'$, so the consequent state ϑ' is obtained by taking the jump $reduce_e@p$ for a continuous place p. The semantic rule is given in Definition 2.12 by

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

We thus know that $fireable_\vartheta(reduce_e@p)$, $\nu' = disc_\vartheta(reduce_e@p)$, and $\nu'_{Lab} = \nu_{Lab}[a_1@p \mapsto u]$ for $u \in supp(Dur(a_1@p, (loc, \nu')))$. Note that since $supp(Dur(a_1@p, (loc, \nu'))) = \{0\}$, we know that $u = 0$ and $\nu'_{Lab} = \nu_{Lab}$.

We start by examining what we can infer about $\tilde{f}^{-1}(\vartheta)$ and $\tilde{f}^{-1}(\vartheta')$.

- For every discrete place $p' \in \mathcal{P}^{disc}$, we have

$$\begin{aligned} m'(p') &= \nu'(m_{p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(reduce_e@p)(m_{p'}) \\ &= reduce_p^{empty}(\nu)(m_{p'}) && \text{by def. of } reduce_e \\ &= \nu(m_{p'}) && \text{by Definition 4.7} \\ &= m(p') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  Therefore, we have that $m' = m$.

- For every continuous place $p' \in \mathcal{P}^{cont}$, we have

$$\begin{aligned} x'(p') &= \nu'(x_{p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(reduce_e@p)(x_{p'}) \\ &= reduce_p^{empty}(\nu)(x_{p'}) && \text{by def. of } reduce_e \\ &= \nu(x_{p'}) && \text{by Definition 4.7} \\ &= x(p') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  So, also $x' = x$.

- For every discrete transition $t \in \mathcal{T}^{disc}$, we have

$$\begin{aligned} c'(t) &= \nu'_{Lab}(a_t) && \text{by def. of } \tilde{f}^{-1} \\ &= \nu_{Lab}(a_t) && \text{by assumption} \\ &= c(t) && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  Therefore, we have that $c' = c$.

- Finally, for $t \in \mathcal{T}^{cont}$ and $p' \in \mathcal{P}^{cont}$ we have

$$\begin{aligned} l'(t)(p') &= \nu'(l_{t,p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(reduce_e@p)(l_{t,p'}) \\ &= reduce_p^{empty}(\nu)(l_{t,p'}) && \text{by def. of } reduce_e \\ &= \begin{cases} \frac{in(p)}{out(p)} \cdot \min\left\{\nu(l_{t,p''}) \mid p'' \in \mathcal{P}^{cont}\right\} & \text{if } p = p' \\ \nu(l_{t,p'}) & \text{else} \end{cases} && \text{by Definition 4.7} \\ &= \begin{cases} \frac{in(p)}{out(p)} \cdot \min\left\{l(t)(p'') \mid p'' \in \mathcal{P}^{cont}\right\} & \text{if } t \in T, p = p' \\ l(t)(p') & \text{else.} \end{cases} && \text{by def. of } \tilde{f}^{-1} \\ &= reduce_p^{empty}(l)(t)(p') && \text{by Definition 3.13} \end{aligned}$$

  So, we have that

$$l' = reduce_p^{empty}(l)$$

We aim to show that $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$, and from the considerations above we know that

$$\tilde{f}^{-1}(\vartheta') = (m', x', c', l') = (m, x, c, reduce_p^{empty}(l)).$$

In the transformation, taking the jump $reduce_e@p$ simulates the reduction step of the rate adaption. Therefore, we are going to show that

$$(m, x, c, l) \xRightarrow{\text{reduce empty } p} (m, x, c, reduce_p^{empty}(l)).$$

The rule as defined in Section 3.2.3 is

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = 0 \quad drift_\sigma(p) < 0}{(m, x, c, l) \xRightarrow{\text{reduce empty } p} (m, x, c, reduce_p^{empty}(l))} \text{ reduce\_empty}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that $x(p) = 0$ and that $drift_\sigma(p) < 0$. We know that $fireable_\vartheta(reduce_e@p)$, so the guard of $reduce_e@p$ is satisfied in ϑ. By definition of the jump in Definition 4.10, we thus know that $\nu(x_p) = 0$ and $drift_\nu(p) < 0$. This directly implies that $x(p) = 0$ by definition of $\tilde{f}^{-1}$ and that $drift_\sigma(p) < 0$ by (6).

In conclusion, all premises hold and we have that

$$\vartheta \xRightarrow{reduce_e@p} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) \xRightarrow{\text{reduce empty } p} \tilde{f}^{-1}(\vartheta').$$

In case p is at its upper boundary case, the argumentation is dual and we can show that

$$\vartheta \xRightarrow{reduce_f@p} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) \xRightarrow{\text{reduce full } p} \tilde{f}^{-1}(\vartheta').$$

Rate Adaption: Reset

Assume that $\vartheta \xRightarrow{reset_e@p} \vartheta'$, so the consequent state ϑ' is obtained by taking the jump $reset_e@p$ for a continuous place p. The semantic rule is given in Definition 2.12 by

$$\frac{e = (id, l, \mu, l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e), (l', \nu')))}{(l, \nu, \nu_{Lab}) \xRightarrow{e} (l', \nu', \nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

We thus know that $fireable_\vartheta(reset_e@p)$, $\nu' = disc_\vartheta(reset_e@p)$, and $\nu'_{Lab} = \nu_{Lab}[a_3@p \mapsto u]$ for $u \in supp(Dur(a_3@p, (loc, \nu')))$. Note that since $supp(Dur(a_3@p, (loc, \nu'))) = \{0\}$, we know that $u = 0$ and $\nu'_{Lab} = \nu_{Lab}$.

We start by examining what we can infer about $\tilde{f}^{-1}(\vartheta)$ and $\tilde{f}^{-1}(\vartheta')$.

- For every discrete place $p' \in \mathcal{P}^{disc}$, we have

$$\begin{aligned} m'(p') &= \nu'(m_{p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(reset_e@p)(m_{p'}) \\ &= reset_p(\nu)(m_{p'}) && \text{by def. of } reset_e \\ &= \nu(m_{p'}) && \text{by Definition 4.9} \\ &= m(p') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  So, $m' = m$.

- For every continuous place $p' \in \mathcal{P}^{cont}$, we have

$$\begin{aligned} x'(p') &= \nu'(x_{p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(reset_e@p)(x_{p'}) \\ &= reset_p(\nu)(x_{p'}) && \text{by def. of } reset_e \\ &= reset_p(\nu)(x_{p'}) \\ &= \nu(x_{p'}) && \text{by Definition 4.9} \\ &= x(p') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  So, also $x' = x$.

- For every discrete transition $t \in \mathcal{T}^{disc}$, we have

$$\begin{aligned} c'(t) &= \nu'_{Lab}(a_t) && \text{by def. of } \tilde{f}^{-1} \\ &= \nu_{Lab}(a_t) && \text{by assumption} \\ &= c(t) && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  So, $c' = c$.

- Finally, for $t \in \mathcal{T}^{cont}$ and $p' \in \mathcal{P}^{cont}$ we have

$$\begin{aligned} l'(t)(p') &= \nu'(l_{t,p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(reset_e@p)(l_{t,p'}) \\ &= reset_p(\nu)(l_{t,p'}) && \text{by def. of } reset_e \\ &= \begin{cases} 1 & \text{if } p = p' \\ \nu(l_{t,p'}) & \text{else} \end{cases} && \text{by Definition 4.9} \\ &= \begin{cases} 1 & \text{if } p = p' \\ l(t)(p') & \text{else} \end{cases} && \text{by def. of } \tilde{f}^{-1} \\ &= reset_p(l)(t)(p') && \text{by Definition 3.15} \end{aligned}$$

  So, we have that $l' = reset_p(l)$.

We aim to show that $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$, and from the considerations above we know that

$$\tilde{f}^{-1}(\vartheta') = (m', x', c', l') = (m, x, c, reset_p(l)).$$

In the transformation, taking the jump $reset_e$ simulates the reset step of the rate adaption algorithm. Therefore, we are going to show that

$$(m, x, c, l) \xRightarrow{\text{reset empty } p} (m, x, c, reset_p(l)).$$

The rule as defined in Section 3.2.3 is

$$\frac{p \in \mathcal{P}^{cont} \quad x(p) = 0 \quad drift_\sigma(p) > 0 \quad restr_\sigma(p)}{(m,x,c,l) \xrightarrow{\text{reset empty } p} (m,x,c,reset_p(l))} \text{ reset\_empty}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that $x(p) = 0$, $drift_\sigma(p) > 0$, and $restr_\sigma(p)$. We know that $fireable_\vartheta(reset_e@p)$, so the guard of $reset_e@p$ is satisfied in ϑ. By definition of the jump in Definition 4.10, we thus know that $\nu(x_p) = 0$, $drift_\nu(p) > 0$, and $restr_\nu(p)$. With the definition of $\tilde{f}^{-1}$, (6), and (4), these conditions directly imply that all premises of the rule hold. In conclusion, we have that

$$\vartheta \xrightarrow{\text{rule}} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) \xrightarrow{\text{reset empty } p} \tilde{f}^{-1}(\vartheta').$$

In case p is at its upper boundary case, the argumentation is dual and we can show that

$$\vartheta \xrightarrow{reset_f@p} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) \xrightarrow{\text{reset full } p} \tilde{f}^{-1}(\vartheta').$$

Rate Adaption: Reset Single

Assume that $\vartheta \xrightarrow{resetSingle@p} \vartheta'$, so the consequent state ϑ' is obtained by taking the jump $resetSingle@p$ for a continuous place p. The semantic rule is given in Definition 2.12 by

$$\frac{e = (id,l,\mu,l') \in Edge \quad fireable_\vartheta(e) \quad \nu' = disc_\vartheta(e) \quad u \in supp(Dur(Proc(e),(l',\nu')))}{(l,\nu,\nu_{Lab}) \xRightarrow{e} (l',\nu',\nu_{Lab}[Proc(e) \mapsto u])} \text{ discrete}$$

We thus know that $fireable_\vartheta(resetSingle@p)$ holds, the new valuation ν' is given by $disc_\vartheta(resetSingle@p)$, and that the labels are updated by $\nu'_{Lab} = \nu_{Lab}[a_5@p \mapsto u]$, i.e., they only change at the entry $a_5@p$. Note that since $supp(Dur(a_5@p,(loc,\nu'))) = \{0\}$, we know that $u = 0$ and $\nu'_{Lab} = \nu_{Lab}$.

We start by examining what we can infer about $\tilde{f}^{-1}(\vartheta)$ and $\tilde{f}^{-1}(\vartheta')$.

- For every discrete place $p' \in \mathcal{P}^{disc}$, we have

$$\begin{aligned} m'(p') &= \nu'(m_{p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(resetSingle@p)(m_{p'}) \\ &= \nu[l_{t,p} \mapsto 1](m_{p'}) && \text{by def. of } resetSingle \\ &= \nu(m_{p'}) \\ &= m(p') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  Therefore, we have that $m' = m$.

- For every continuous place $p' \in \mathcal{P}^{cont}$, we have

$$\begin{aligned} x'(p') &= \nu'(x_{p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(resetSingle@p)(x_{p'}) \\ &= \nu[l_{t,p} \mapsto 1](x_{p'}) && \text{by def. of } resetSingle \\ &= \nu(x_{p'}) \\ &= x(p') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  So, also $x' = x$.

- For every discrete transition $t' \in \mathcal{T}^{disc}$, we have

$$\begin{aligned} c'(t') &= \nu'_{Lab}(a_{t'}) && \text{by def. of } \tilde{f}^{-1} \\ &= \nu_{Lab}(a_{t'}) && \text{by assumption} \\ &= c(t') && \text{by def. of } \tilde{f}^{-1} \end{aligned}$$

  So, $c' = c$.

- Finally, for $t' \in \mathcal{T}^{cont}$ and $p' \in \mathcal{P}^{cont}$ we have

$$\begin{aligned} l'(t')(p') &= \nu'(l_{t',p'}) && \text{by def. of } \tilde{f}^{-1} \\ &= disc_\vartheta(resetSingle@p)(l_{t',p'}) \\ &= \nu[l_{t,p} \mapsto 1](l_{t',p'}) && \text{by def. of } resetSingle \\ &= \begin{cases} 1 & \text{if } t' = t, p' = p \\ \nu(l_{t',p'}) & \text{else} \end{cases} \\ &= \begin{cases} 1 & \text{if } t' = t, p' = p \\ l(t')(p') & \text{else} \end{cases} && \text{by def. of } \tilde{f}^{-1} \\ &= l[t, p \mapsto 1](t')(p') \end{aligned}$$

  So, we have that

$$l' = l[t, p \mapsto 1]$$

We aim to show that $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$, and from the considerations above we know that

$$\tilde{f}^{-1}(\vartheta') = (m', x', c', l') = (m, x, c, l[t, p \mapsto 1]).$$

In the transformation, the jump *resetSingle* simulates the maintenance of the restriction list. Therefore, we are going to show that

$$(m, x, c, l) \xRightarrow{\text{reset } t,\, p} (m, x, c, l[t, p \mapsto 1]).$$

The rule as defined in Section 3.2.3 is

$$\frac{t \in \mathcal{T}^{cont} \quad \neg fireable_\sigma(t) \quad p \in \mathcal{P}^{cont} \quad l(t)(p) < 1}{(m,x,c,l) \xrightarrow{\text{reset } t,p} (m,x,c,l[t,p \mapsto 1])} \text{ reset\_single}$$

With the above observations in mind, we now prove that this rule is indeed applicable. This concretely means that we have to show that there exists a transition $t \in \mathcal{T}^{cont}$ such that $\neg fireable_\sigma(t)$ and $l(t)(p) < 1$. We know that $fireable_\vartheta(resetSingle@p)$, so the guard of $resetSingle@p$ is satisfied in ϑ. By definition of the jump in Definition 4.10, we thus know that there exists a transition $t \in \mathcal{T}^{cont}$ with $\neg conc_\nu(t)$ and $\nu(l_{t,p}) < 1$. We can follow that $\neg fireable_\sigma(t)$ by (1) and the fact that being fireable equals having concession for continuous transitions, and that $l(t)(p) < 1$ by definition of $\tilde{f}^{-1}$.

In conclusion, all premises hold and we have that

$$\vartheta \xrightarrow{resetSingle@p} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) \xrightarrow{\text{reset } t,p} \tilde{f}^{-1}(\vartheta').$$

Passing of Time

Assume that $\vartheta \xrightarrow{\tau,f} \vartheta'$, so the consequent state ϑ' is obtained by letting τ time units pass following an activity $f \in Act(loc)$. The semantic rule is given in Definition 2.12 by

$$\frac{f \in Act(l) \quad \tau \in \mathbb{R}_{>0} \quad \nu' = time^f_\vartheta(\tau) \quad \nu'_{Lab} = clocks^f_\vartheta(\tau)}{(l,\nu,\nu_{Lab}) \xrightarrow{\tau,f} (l,\nu',\nu'_{Lab})} \text{ time}$$

We thus know that $\nu' = time^f_\vartheta(\tau)$ and $\nu'_{Lab} = clocks^f_\vartheta(\tau)$. Since $f \in Act(loc)$, we know that f must satisfy

- $\dot{x}_p = drift_\nu(p)$ for all $p \in \mathcal{P}^{cont}$ and
- $\dot{y} = 0$ for all other $y \in Var^{\mathcal{H}}$.

From the fact that $\nu' = time^f_\vartheta(\tau)$, we know that $f(0) = \nu$ and $f(\tau) = \nu'$. Therefore, ν' only changes the variables x_p for continuous places by their drift. Formally, this means that

$$\nu'(x_p) = \nu(x_p) + \tau \cdot drift_\nu(p) \quad \text{and} \quad \nu'(y) = \nu(y) \text{ for } y \neq x_p.$$

The fact that $\nu'_{Lab} = clocks^f_\vartheta(\tau)$ intuitively means that all jumps corresponding to a label are either enabled in the complete time interval or not at all, and that the clocks are decreased accordingly. The formal description is given in Definition 2.11 and states that for all $r \in Lab$, one of the following is true:

- There exists some $e = (id, l, \mu, l') \in Edge$ with $r = Proc(e)$ such that

1. $\forall \tau' \in (0, \tau).\ \exists \nu'' \in \mathcal{V}.\ \nu'' = time^{f}_{\vartheta}(\tau') \wedge enabled_{\vartheta''}(e)$ for $\vartheta'' = (l, \nu'', \nu_{Lab})$, and
2. $\nu''_{Lab}(r) = \nu_{Lab}(r) - \tau \geq 0$,

- or for all $e = (id, l, \mu, l') \in Edge$ with $r = Proc(e)$ it holds that
  1. $\forall \tau' \in (0, \tau).\ \neg\exists \nu'' \in \mathcal{V}.\ \nu'' = time^{f}_{\vartheta}(\tau') \wedge enabled_{\vartheta''}(e)$ for $\vartheta'' = (l, \nu'', \nu_{Lab})$, and
  2. $\nu''_{Lab}(r) = \nu_{Lab}(r)$.

As we did in Appendix B.2.2, we will show that labels a_t for deterministic transitions with concession fall into the first category, and all other labels into the second. For this purpose, we define two sets

$$T_1 = \left\{ a_t \in Lab \;\middle|\; t \in \mathcal{T}^{disc} \text{ has concession in } (0, \tau) \right\} \quad \text{and} \quad T_2 = Lab \backslash T_1.$$

Note that T_1 corresponds to the set $conc^{disc}_{\tilde{f}^{-1}(\vartheta),\tau}$ from Definition 3.17. We argue that T_1 contains the labels in the first category and T_2 those in the second. For this, first note that all labels a that are not of the form a_t for a discrete transition t are always 0. In particular, this implies that $\nu_{Lab}(a) - \tau < 0$ for all $\tau > 0$. Therefore, all such labels must fall into the second category.

For labels a_t, we distinguish between transitions that have concession and those that do not. Recall that $enabled_{\vartheta'}(jump_t)$ is with (2) equal to $conc_{\tilde{f}^{-1}(\vartheta')}(t)$. Therefore, all jumps in T_1 that have concession in all intermediate states are in the first category, and all others are in the second.

To conclude, we have that $\nu'_{Lab}(a_t) = \nu_{Lab}(a_t) - \tau$ for all $t \in T_1$ and $\nu'_{Lab}(a) = \nu_{Lab}(a)$ for all other labels.

We continue with examining what we can infer about $\tilde{f}^{-1}(\vartheta)$ and $\tilde{f}^{-1}(\vartheta')$.

- We know that by the above reasoning, f does not modify the marking, so we have $\nu'(m_p) = \nu(m_p)$ for all $p \in \mathcal{P}^{disc}$. By definition of $\tilde{f}^{-1}$, this implies that $m' = m$.
- By the above considerations, we know that $\nu'(x_p) = \nu(x_p) + \tau \cdot drift_{\nu}(p)$. By definition of $\tilde{f}^{-1}$ and (6), this implies that $x'(p) = x(p) + \tau \cdot drift_{\tilde{f}^{-1}(\vartheta)}(p)$.

  Now, we need to take into account that there is a slight difference in the requirements for the passing of time. In HPnGs, we require that all continuous transitions to be either continuously fireable or not in the full time span. This is not required in the transformed HPnG, because the changes in the level of places are described with activity functions. Therefore, we cannot be sure that the set of fireable continuous transitions is stable, i.e., does not change, within τ. For this purpose, we split the time interval into several smaller intervals $\tau = \tau_1 + \cdots + \tau_n$, where the enabling status of (at least) one continuous changes at time τ_i and

within an interval the statuses do not change. Let $\vartheta_1 = \vartheta$ and $\vartheta_i \overset{\tau_i}{\Longrightarrow} \vartheta_{i+1}$ be the intermediate states, so $\vartheta_{n+1} = \vartheta'$.

Recall that $fireable^{cont}_{\tilde{f}^{-1}(\vartheta),\tau}$ are the continuous transitions enabled in the time span τ (see Definition 3.16). Using the splitting as above, we get that $fireable^{cont}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}$ is a stable set for all $i \in \{1,\dots,n\}$. Denote $\tilde{f}^{-1}(\vartheta_i) = (m_i, x_i, c_i, l_i)$. Then, we have for $i \in \{2,\dots,n+1\}$ that

$$x_i(p) = x_{i-1}(p) + \tau_i \cdot drift_{\tilde{f}^{-1}(\vartheta_i)}(p) = fire^{\tau_i}_{fireable^{cont}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(x)(p).$$

The final level of place p can be computed as

$$x_{n+1}(p) = x'(p) = x(p) + \sum_{i=1}^{n} \tau_i \cdot drift_{\tilde{f}^{-1}(\vartheta_i)}(p).$$

For every intermediate state, we thus have $x_i = fire^{\tau_i}_{fireable^{cont}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(x)$.

- As argued above, we have $\nu'_{Lab}(a_t) = \nu_{Lab}(a_t) - \tau$ for all $t \in T_1$ and $\nu'_{Lab}(a) = \nu_{Lab}(a)$ otherwise. Since $T_1 = conc^{disc}_{\tilde{f}^{-1}(\vartheta),\tau}$ and by definition of $\tilde{f}^{-1}$, we have that $c' = evolve^{\tau}_{conc^{disc}_{\tilde{f}^{-1}(\vartheta),\tau}}(c)$. In contrast to the set of fireable continuous transitions, the set of discrete transitions with concession is stable in the full time span. Therefore, we do not have to split the time span here.
- We know that f does not modify the list of the restrictions, so we have $\nu'(l_{t,p}) = \nu(l_{t,p})$ for all $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$. By definition of $\tilde{f}^{-1}$, this implies that $l' = l$.

We aim to show that $\tilde{f}^{-1}(\vartheta) \Rightarrow \tilde{f}^{-1}(\vartheta')$. As was indicated above, we will do so by taking several time steps using the splitting of τ and show that

$$\tilde{f}^{-1}(\vartheta) = \tilde{f}^{-1}(\vartheta_1) \overset{\tau_1}{\Longrightarrow} \tilde{f}^{-1}(\vartheta_2) \overset{\tau_2}{\Longrightarrow} \dots \overset{\tau_n}{\Longrightarrow} \tilde{f}^{-1}(\vartheta_{n+1}) = \tilde{f}^{-1}(\vartheta').$$

From the considerations above, we know that

$$\tilde{f}^{-1}(\vartheta_i) = (m, fire^{\tau_i}_{fireable^{cont}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(x), evolve^{\tau_i}_{conc^{disc}_{\tilde{f}^{-1}(\vartheta),\tau_i}}(c), l)$$

for $i \in \{2,\dots,n+1\}$. The rule for a time step as defined in Section 3.2.3 is given by

$$\frac{\tau \in \mathbb{R}_{>0} \quad safe_{\sigma}(\tau)}{(m,x,c,l) \overset{\tau}{\Longrightarrow} (m, fire^{\tau}_{fireable^{cont}_{\sigma,\tau}}(x), evolve^{\tau}_{conc^{disc}_{\sigma,\tau}}(c), l)} \text{ time}$$

With the above observations in mind, we now prove that this rule is indeed applicable for each time interval. This concretely means that we have to show that $safe_{\tilde{f}^{-1}(\vartheta_i)}(\tau_i)$ holds for every $i \in \{2,\dots,n+1\}$.

In Definition 3.18, this was defined to hold if for all $\tau' \in (0, \tau_i)$ and corresponding intermediate states $\sigma' = (m, fire^{\tau'}_{fireable^{cont}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(x), evolve^{\tau'}_{conc^{disc}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(c), l)$ the following five conditions hold. First, note that we can conclude from the reasoning above that the intermediate states in $\mathcal{A}^{\mathcal{H}}$ and $\mathcal{H}$ coincide, i.e., we have that $\tilde{f}^{-1}(\vartheta'') = \sigma'$ for all intermediate states ϑ'' with $\nu'' = time^f_\vartheta(\tau')$ and $\nu''_{Lab} = clocks^f_\vartheta(\tau')$. Also, we will only need the splitting of the time span for proving the first part of the firth condition. All other conditions hold regardless of that splitting for the full time span.

1. $\neg fireable_{\sigma'}(t)$ for all $t \in \mathcal{T}^{disc}$

   Since we know that $\nu'_{Lab} = clocks^f_\vartheta(\tau)$ holds, all jumps either are enabled in the full time span and their clocks reach zero earliest at time τ, or they are not enabled in the full time span. This in particular holds for all jumps $jump_t$. From (2), we know that t having concession corresponds to $jump_t$ being enabled in states related by $\tilde{f}$. Since $fireable_{\sigma'}(t)$ only holds if t has concession and their clock is zero, we know that this is not fulfilled in the time span.

2. $x'(p) = 0 \implies drift_{\sigma'}(p) = 0 \vee (drift_{\sigma'}(p) > 0 \wedge \neg restr_{\sigma'}(p))$ for all $p \in \mathcal{P}^{cont}$

   We already know that all jumps that are not of the form $jump_t$ for a transition t are in T_2, i.e., are not enabled in any intermediate state ϑ'. This in particular means that the jump $reduce_e@p \notin T_2$ for all places $p \in \mathcal{P}^{cont}$ is not enabled. By definition of this jump in Definition 4.10, we know that this is the case if either $\nu'(x_p) > 0$ or $drift_{\nu'}(p) \geq 0$. With (6) and the definition of $\tilde{f}^{-1}$, this implies that $x'(p) > 0$ or $drift_{\sigma'}(p) \geq 0$.

   Additionally, we know that the jump $reset_e@p \notin T_2$ for all places $p \in \mathcal{P}^{cont}$ is not enabled. By definition of this jump in Definition 4.10, we know that this is the case if $\nu(x_p) \neq 0$, $\neg restr_\nu(p)$, or $drift_\nu(p) \leq 0$. With (6), (4), and the definition of $\tilde{f}^{-1}$, this implies that $x'(p) > 0$, $\neg restr_{\sigma'}(p)$, or $drift_{\sigma'}(p) \leq 0$.

   In combination with the earlier properties, we get that $x'(p) = 0$ implies that $drift_{\sigma'}(p) = 0$, or $drift_{\sigma'}(p) > 0$ and $\neg restr_{\sigma'}(p))$.

3. $x'(p) = \Phi^{\mathcal{P}}_{ub}(p) \implies drift_{\sigma'}(p) = 0 \vee (drift_{\sigma'}(p) < 0 \wedge \neg restr_{\sigma'}(p))$ for all $p \in \mathcal{P}^{cont}$

   The arguments for this case are dual to those above: We can make use of the fact that $reduce_f@p \notin T_2$ and $reset_f@p \notin T_2$ are not enabled.

4. $\neg fireable_{\sigma'}(t) \implies l(t)(p) = 1$ for all $t \in \mathcal{T}^{cont}, p \in \mathcal{P}^{cont}$

   Similar to the cases above, we now use that the jump $resetSingle@p \in T_2$ does not become enabled. By Definition 4.10, this means that for all transitions $t \in \mathcal{T}^{cont}$, we have either $conc_{\nu'}(t)$ or $\nu'(l_{t,p}) = 1$. With (1) and the definition of $\tilde{f}^{-1}$, this is equal to $conc_{\sigma'}(t)$ or $l(t)(p) = 1$ for all transitions $t \in \mathcal{T}^{cont}$ and places $p \in \mathcal{P}^{cont}$. Since the notions of having concession and being fireable coincide for continuous

transitions, this is logically equivalent to $\neg \mathit{fireable}_{\sigma'}(t) \implies l(t)(p) = 1$ and the claim is proven.

5. for all $\tau'' \in (0, \tau)$ and corresponding intermediate states $\sigma'' = (m, x'', c'', l) = (m, \mathit{fire}^{\tau''}_{\mathit{fireable}^{cont}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(x), \mathit{evolve}^{\tau''}_{\mathit{conc}^{disc}_{\tilde{f}^{-1}(\vartheta_i),\tau_i}}(c), l)$ we have

   a) $\mathit{fireable}_{\sigma''}(t) = \mathit{fireable}_{\sigma'}(t)$ for all $t \in \mathcal{T}^{cont}$

      Now, we need the fact that we split the time interval such that the continuous transitions remain fireable in each interval. In particular, the splitting was chosen in such a way that for state in the interval (τ_i, τ_{i+1}), the fireability of a continuous transition does not change. This exactly corresponds to the above condition.

   b) $\mathit{conc}_{\sigma''}(t) = \mathit{conc}_{\sigma'}(t)$ for all $t \in \mathcal{T}^{disc}$

      As was elaborated earlier, the enabling status of jump_t does not change in the time interval, as we have $\nu'_{Lab} = \mathit{clocks}^f_\vartheta(\tau)$. Since jump_t is enabled exactly if t has concession, the above condition is implied.

In conclusion, all premises hold and we have that

$$\vartheta \overset{\tau, f}{\Longrightarrow} \vartheta' \quad \text{implies} \quad \tilde{f}^{-1}(\vartheta) = \tilde{f}^{-1}(\vartheta_1) \overset{\tau_1}{\Longrightarrow} \tilde{f}^{-1}(\vartheta_2) \overset{\tau_2}{\Longrightarrow} \ldots \overset{\tau_n}{\Longrightarrow} \tilde{f}^{-1}(\vartheta_{n+1}) = \tilde{f}^{-1}(\vartheta')$$

for a suitable splitting of τ.

Conclusion. We have now shown that for $\sigma, \sigma' \in \Sigma^{\mathcal{H}}$ we have

$$\sigma \Rightarrow \sigma' \quad \text{iff} \quad \tilde{f}(\sigma) \Rightarrow \tilde{f}(\sigma'),$$

which finally concludes the proof of the correctness of the transformation as formulated in Theorem 4.1.

Bibliography

[AA97] M. Allam and H. Alla. "Modelling production systems by hybrid automata and hybrid Petri nets". In: *IFAC Proceedings Volumes* 30.6 (1997), pp. 343–348.

[AA98] M. Allam and H. Alla. "From hybrid Petri nets to hybrid automata". In: *Journal européen des systèmes automatisés* 32.9-10 (1998), pp. 1165–1185.

[Aba+07] A. Abate, S. Amin, M. Prandini, J. Lygeros, and S. Sastry. "Computational approaches to reachability analysis of stochastic hybrid systems". In: *International Workshop on Hybrid Systems: Computation and Control*. Springer. 2007, pp. 4–17.

[ACB84] M. Ajmone Marsan, G. Conte, and G. Balbo. "A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems". In: *ACM Transactions on Computer Systems (TOCS)* 2.2 (1984), pp. 93–122.

[Alt15] M. Althoff. "An introduction to CORA 2015". In: *Proceedings of the workshop on applied verification for continuous and hybrid systems*. 2015, pp. 120–151.

[CÁS12] X. Chen, E. Ábrahám, and S. Sankaranarayanan. "Taylor model flowpipe construction for non-linear hybrid systems". In: *2012 IEEE 33rd Real-Time Systems Symposium*. IEEE. 2012, pp. 183–192.

[DA01] R. David and H. Alla. "On hybrid Petri nets". In: *Discrete Event Dynamic Systems* 11.1 (2001), pp. 9–40.

[DA10] R. David and H. Alla. *Discrete, continuous, and hybrid Petri nets*. Vol. 1. Springer, 2010.

[Eis+13] C. Eisentraut, H. Hermanns, J.-P. Katoen, and L. Zhang. "A semantics for every GSPN". In: *International Conference on Applications and Theory of Petri Nets and Concurrency*. Springer. 2013, pp. 90–109.

[Frä+11] M. Fränzle, E. M. Hahn, H. Hermanns, N. Wolovick, and L. Zhang. "Measurability and safety verification for stochastic hybrid systems". In: *Proceedings of the 14th international conference on Hybrid systems: computation and control*. 2011, pp. 43–52.

[Fre+11] G. Frehse, C. L. Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. "SpaceEx: Scalable verification of hybrid systems". In: *International Conference on Computer Aided Verification*. Springer. 2011, pp. 379–395.

[GA18] L. Ghomri and H. Alla. "Continuous Petri nets and hybrid automata: Two bisimilar models for the simulation of positive systems". In: *International Journal of Simulation and Process Modelling* 13.1 (2018), pp. 24–34.

[Ger22] C. Gerlach. "Compositional modeling of stochastic hybrid systems". Master's thesis. RWTH Aachen University, 2022.

[Gha17] H. Ghasemieh. "Analysis of hybrid Petri nets with random discrete events". Dissertation. University of Twente, 2017.

[GR10] M. Gribaudo and A. Remke. "Hybrid Petri nets with general one-shot transitions for dependability evaluation of fluid critical infrastructures". In: *2010 IEEE 12th International Symposium on High Assurance Systems Engineering*. IEEE. 2010, pp. 84–93.

[GR16] M. Gribaudo and A. Remke. "Hybrid Petri nets with general one-shot transitions". In: *Performance Evaluation* 105 (2016), pp. 22–50.

[GRH16] H. Ghasemieh, A. Remke, and B. R. Haverkort. "Survivability analysis of a sewage treatment facility using hybrid Petri nets". In: *Performance evaluation* 97 (2016), pp. 36–56.

[Hah+13] E. M. Hahn, A. Hartmanns, H. Hermanns, and J.-P. Katoen. "A compositional modelling and analysis framework for stochastic hybrid systems". In: *Formal Methods in System Design* 43.2 (2013), pp. 191–232.

[HH14] A. Hartmanns and H. Hermanns. "The Modest Toolset: An integrated environment for quantitative modelling and verification". In: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer. 2014, pp. 593–598.

[HNR20] J. Hüls, H. Niehaus, and A. Remke. "HPNMG: A C++ tool for model checking hybrid Petri nets with general transitions". In: *NASA Formal Methods Symposium*. Springer. 2020, pp. 369–378.

[HR19] J. Hüls and A. Remke. "Model checking HPnGs in multiple dimensions: Representing state sets as convex polytopes". In: *International Conference on Formal Techniques for Distributed Objects, Components, and Systems*. Springer. 2019, pp. 148–166.

[Hül+21] J. Hüls, C. Pilch, P. Schinke, H. Niehaus, J. Delicaris, and A. Remke. "State-space construction of hybrid Petri nets with multiple stochastic firings". In: *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 31.3 (2021), pp. 1–37.

[JP09] A. A. Julius and G. J. Pappas. "Approximations of stochastic hybrid systems". In: *IEEE Transactions on Automatic Control* 54.6 (2009), pp. 1193–1203.

[PER17] C. Pilch, F. Edenfeld, and A. Remke. "Hypeg: Statistical model checking for hybrid Petri nets: Tool paper". In: *Proceedings of the 11th EAI International Conference on Performance Evaluation Methodologies and Tools*. 2017, pp. 186–191.

[Pil+20] C. Pilch, M. Krause, A. Remke, and E. Ábrahám. "A transformation of hybrid Petri nets with stochastic firings into a subclass of stochastic hybrid automata". In: *NASA Formal Methods Symposium*. Springer. 2020, pp. 381–400.

[Pol+03] G. Pola, M.-L. Bujorianu, J. Lygeros, and M. D. Di Benedetto. "Stochastic hybrid models: An overview". In: *IFAC Proceedings Volumes* 36.6 (2003), pp. 45–50.

[PR17] C. Pilch and A. Remke. "Statistical model checking for hybrid Petri nets with multiple general transitions". In: *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE. 2017, pp. 475–486.

[SM00] W. H. Sanders and J. F. Meyer. "Stochastic activity networks: Formal definitions and concepts". In: *School organized by the European Educational Forum*. Springer. 2000, pp. 315–343.

[TK93] K. S. Trivedi and V. G. Kulkarni. "FSPNs: Fluid stochastic Petri nets". In: *International Conference on Application and Theory of Petri Nets*. Springer. 1993, pp. 24–31.

Index

Definitions

Theorems

Examples