Partial bisimulation

Systems Engineering Group
Department of Mechanical Engineering
Eindhoven University of Technology
PO Box 513
5600 MB Eindhoven
The Netherlands
http://se.wtb.tue.nl/

SE Report: Nr. 2010-04

Partial Bisimulation

J.C.M. Baeten D.A. van Beek B. Luttik
J. Markovski J.E. Rooda [1]

ISSN: 1872-1567

SE Report: Nr. 2010-04
Eindhoven, April 2010
SE Reports are available via http://se.wtb.tue.nl/sereports

Abstract

We investigate partial bisimulation preorder, a behavioral preorder that is coarser than bisimulation equivalence and finer than simulation preorder. Partial bisimulation preorder is parameterized with a subset of actions; if two processes are related, then transitions labeled with an action in this subset need to be simulated in both directions, whereas transitions labeled with an action outside the subset need to be simulated in one direction only. The parameter is employed to define a notion of controllability, and discuss the supervisory control synthesis problem in a process-theoretic setting. Our notion of controllability improves on the current definition of controllability of nondeterministic discrete-event systems, which is given in terms of traces and/or refusal sets. We present a sound and ground-complete axiomatization of the partial bisimulation preorder and a modal characterization. We also provide for a partitioning algorithm that computes the partial bisimulation equivalence, and can serve as a minimization procedure in a supervisory control setting. The algorithm is based on partition pairs and it employs Paige-Tarjan optimization, which makes it suitable as a basis for an efficient algorithm for simulation minimization as well.

1 Introduction

To keep products competitive, producers need to optimize development costs and minimize time-to-market, while satisfying the ever-changing market demands for higher quality, better performance, new functionalities, improved safety, and ease of use. This puts high demands on the development of control software. Traditionally, software engineers write control software based on specification documents that contain informal requirements. This is a time-consuming error-prone process, since the requirements are often ambiguous and, moreover, they constantly change during the product development. This issue in control software design gave rise to supervisory control theory of discrete-event systems [27, 8], where high-level supervisory controllers are synthesized automatically based upon formal models of the hardware and control requirements.

The supervisory controller observes the discrete-event behavior of the machine by receiving control signals from ongoing activities, typically from sensors inside the machine. Based upon these signals it makes a decision which activities the machine is allowed to carry out and sends back control signals to the actuators, which control the hardware. This is known as a *feedback loop*. Under the assumption that the supervisory controller can react sufficiently fast on every input from the machine, one can model the feedback loop as a pair of synchronizing processes. The model of the machine, referred to as *plant*, is restricted by the model of the controller, referred to as *supervisor*. Originally, the plant is modeled as a set of observable traces of events, given as a set of synchronizing automata, whose joint recognized language corresponds to the observed traces. The events are split into *controllable events*, which can be disabled by the supervisor in the synchronous composition, and *uncontrollable events*, which must always be allowed. Thus, the supervisor must always comply with the plant by synchronizing with all uncontrollable events. The *control requirements* specify allowed behavior as sequences of events, again modeled by automata, leading to event-based supervisory control theory [27, 8].

1.1 Controllability

In this paper, we model the feedback loop in a process algebraic setting, revisiting the basic notions in supervisory control theory. The central notion in supervisory control theory is the property of *controllability*. It gives sufficient and necessary conditions when given a plant and control requirements, there exists a supervisor for the plant such that the control requirements are satisfied.

We introduce some preliminary notions of language theory. Let $\mathcal{A} = \mathcal{C} \cup \mathcal{U}$ be the set of events that can be observed in the plant, with $\mathcal{C}$ being the set of controllable events and $\mathcal{U}$ the set of uncontrollable events, such that $\mathcal{C} \cap \mathcal{U} = \emptyset$. We form traces and languages in a standard manner, i.e., $t \in \mathcal{A}^*$ is a trace and $L \subseteq \mathcal{A}^*$ is a language, where $\mathcal{A}^* \triangleq \{a_1 a_2 \dots a_n \mid a_i \in \mathcal{A} \text{ for } 0 \leq i \leq n, n \in \mathbb{N}\}$ and ε is the unique empty trace $a_1 \dots a_n$ for $n = 0$. By $t{\cdot}t'$ we denote the concatenation of the traces $t, t' \in \mathcal{A}^*$ and by $L{\cdot}L' \triangleq \{t{\cdot}t' \mid t \in L, t' \in L'\}$ the concatenation of languages. We say that a language is prefix closed if $L = \overline{L}$, where $\overline{L} \triangleq \{t \mid t{\cdot}t' \in L\}$. Suppose that $P = (\mathcal{S}, \mathcal{A}, \longmapsto, s_0)$ is a standard discrete-event automaton, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ set of events, $\longmapsto \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ the transition relation, and s_0 an initial state. We define $\longmapsto_* \in \mathcal{S} \times \mathcal{A}^* \times \mathcal{S}$ as $s \overset{\varepsilon}{\longmapsto}_* s$ for all $s \in \mathcal{S}$, and $s \overset{at}{\longmapsto}_* s'$ for $a \in \mathcal{A}$ and $t \in \mathcal{A}^*$, if there exists $s'' \in \mathcal{S}$ such that $s \overset{a}{\longmapsto} s'' \overset{t}{\longmapsto}_* s'$. By $s \overset{t}{\longmapsto}_*$ we denote that there exists $s' \in \mathcal{S}$ such that $s \overset{t}{\longmapsto}_* s'$. Now, the recognized prefix-closed language of automaton $P = (\mathcal{S}, \mathcal{A}, \longmapsto, s_0)$ is given by $L(P) \triangleq \{t \mid s_0 \overset{t}{\longmapsto}_* \}$. By $P_1 \mid P_2 \triangleq (\mathcal{S}_1 \times \mathcal{S}_2, \mathcal{A}, \longmapsto, (s_1, s_2))$ we denote the synchronous parallel composition of $P_1 = (\mathcal{S}_1, \mathcal{A}, \longmapsto_1, s_1)$ and $P_2 = (\mathcal{S}_2, \mathcal{A}, \longmapsto_2, s_2)$,

where $(s', s'') \overset{a}{\longmapsto} (\overline{s}', \overline{s}'')$ if $s' \overset{a}{\longrightarrow}_1 \overline{s}'$ and $s' \overset{a}{\longrightarrow}_1 \overline{s}'$ for $s', \overline{s}' \in \mathcal{S}_1$, $s'', \overline{s}'' \in \mathcal{S}_2$, and $a \in \mathcal{A}$. We have that $L(P_1 \mid P_2) = L(P_1) \cap L(P_2)$.

Now, we can define the property of controllability for prefixed closed languages. Suppose that the plant is given by automaton P and the control requirements by R. An automaton S is a supervisor for P and R if $L(P \mid S) = L(R)$, where we refer to $P \mid S$ as the *supervised plant*. We ensure that S does not disable uncontrollable events by requesting that R is *controllable* with respect to P, expressed as $L(R)\cdot\mathcal{U} \cap L(P) = L(R)$ [27, 8]. Controllability is interpreted as follows. If we observe a desired trace in the plant followed by an uncontrollable event, then the control requirements cannot request that this event should be disabled after allowing that trace, as the supervisor does not have control over uncontrollable events. In this case, one can guarantee existence of a supervisor, such that by restricting the plant we achieve the desired controlled behavior. If strict equality is not possible, one can find a maximal supervisor with respect to inclusion, relaxing the requirements to $L(R)\cdot\mathcal{U} \cap L(P) \subseteq L(R)$. One can directly observe that language controllability is not intended for nondeterministic systems. Initial investigations in supervisory synthesis considered additional properties of $P \mid S$, e.g., existence of deadlock or livelock, and suitable notions of controllability that prevent these blocking properties. To this end, marked (or terminal) states are added to the automata to specify nonblocking behavior. In this paper, we do not consider blocking behavior, but we make preparations for future work by incorporating successful termination options in the process theory [2]. Partial observability is another important property, where the assumption is that some events are hidden from the supervisor, e.g., due to lack of sensors [8]. Nonetheless, the supervisory controller must synchronize with the plant on unobservable events as well in order to achieve the desired behavior.

1.2 Related Work

In a way partial observability introduces nondeterminism in supervisory control theory. Note that nondeterministic automata are not disallowed in [27], but they still have semantics in terms of accepted languages. Nondeterminism enables ease of modeling and provides for abstract specifications among else [2]. However, it also introduces a lot of problems, since the original notion of controllability is given in terms of traces. This spawned a large number of investigations into the supervisory control of nondeterministic discrete-event systems.

We briefly review and comment on previous work, most of which deals with nondeterministic plants and nondeterministic control requirements. In general, the supervisor is required to be deterministic, as it is supposed to give feedback to the plant, i.e., send unambiguous control signals. An exception is [10], and references therein, where nondeterministic supervisors are considered, but under strong structural restrictions that require the supervisor to have the same ready sets for uncontrollable events following the same traces. State controllability is one notion tailored for the nondeterministic setting [10, 29] that requires all states of the control requirements reachable by a given trace to enable all uncontrollable events enabled in the plant by following the same trace. Denote by $\mathrm{E}(s,t) \triangleq \{a \in \mathcal{A} \mid s \overset{t}{\longmapsto}_* s' \overset{a}{\longmapsto}_*\}$ the enabled events at all states reachable from s by the trace t. Then, a supervisor S with an initial state s_S is state controllable with respect to a plant P with an initial state s_P, if for all $t \in L(P \mid S)$ it holds that $\mathrm{E}(s_P, t) \cap \mathcal{U} \subseteq \mathrm{E}(s_S, t)$. Note that state controllability becomes standard controllability in the deterministic case. However, it is quite a restrictive notion and, as remarked in [10], some plants are not state controllable with respect to itself, even though a trivial supervisor that enables all events always exists. Other works tackle nondeterministic systems as a set of deterministic systems, by requiring controllability of all underlying deterministic systems to induce controllability of the nondeterministic system [25]. A proposal to replace nondeterminism with a choice between unobservable events is given in [17]. State controllability seems to originate from partial observability as well.

The earliest idea to apply process theory to supervisory synthesis is given in [18], where failure trajectories are employed and a CSP-like axiomatization of a prioritized synchronization operator is given. Failure trajectories are extensions of failure semantics on whole traces, supporting compositionality of the specialized synchronization that is employed to define controllability [18]. It is tailored to model the plant-supervisor communication and ensures that the supervisor cannot disable uncontrollable events. Followup works [17, 19] focus on deepening the understanding of the failure trajectories model and the prioritized synchronization. An alternative path is taken in [23], where instead of a new operator, a refinement relation $\ll$ based on failure semantics characterizes nondeterministic supervised behavior. For the automata P_1 and P_2 from above, $P_1 \ll P_2$ holds, if $L(P_1) \subseteq L(P_2)$, and for all $t \in L(P_1)$ it holds that $\mathcal{A} \setminus \mathrm{E}(s_1, t) \subseteq \mathcal{A} \setminus \mathrm{E}(s_2, t)$, where $\mathcal{A} \setminus \mathrm{E}(s_1, t)$ and $\mathcal{A} \setminus \mathrm{E}(s_2, t)$ are the refusal sets of all states reachable in P_1 and P_2, respectively, following a trace t. Now, in addition to imposing language controllability [23] requires that $P \mid S \ll R$ as well. In [29, 20] the refinement $\ll$ is given in terms of bisimulation and simulation, respectively, relying on the notion of state controllability.

1.3 Motivation and Contributions

A coalgebraic approach to supervisory control theory introduced *partial bisimulation* as a suitable behavioral relation to define controllability [28]. In essence, it suggests that controllable events should be simulated, whereas uncontrollable events should be bisimulated, hence the term partial bisimulation. It serves as a refinement relation between the supervised plant and the control requirements, similar to the approach of [23], but for bisimulation semantics. Even though some research suggests that refinements for failure and bisimulation semantics have mostly the same properties [9], we consider bisimulation as a more appropriate notion to capture nondeterministic behavior of uncontrollable transitions [2, 14]. The refinements in failure semantics are given in terms of traces and inclusion of refusal sets, as in [23], whereas our notion of refinement that characterizes controllability is in terms of simulation-like relations.

Partial bisimulation is closely related to the notion of modal transition systems [21], where from each state there are so-called may and must transitions, corresponding to controllable (simulated) and uncontrollable (bisimulated) transitions in the context of this paper. The problem of synthesizing a supervisor can also be seen as solving a process algebraic equation in the modal transition systems realm [22]. However, refinement by partial bisimulation is a special type of modal refinement, where the labels of the may and must transitions are fixed, admitting elegant process algebraic characterization. Finally, there exist efficient partitioning algorithms for minimization by bisimulation and simulation, which are actually already employed in the deterministic setting to optimize the supervisor synthesis by imposing bisimulation over uncontrollable events [6].

The contributions of this paper are as follows. We give a sound and ground-complete axiomatization, and a modal characterization of the preorder induced by the notion of partial bisimulation and we show some interesting properties of the induced equivalence. We define a notion of controllability using the partial bisimilarity preorder as a refinement between the supervised plant and the control requirements and we characterize the existence of a deterministic supervisor. The partial bisimilarity equivalence serves as a minimization procedure for the plant that preserves controllability, a notion lacking in previous work. We develop a partitioning algorithm for computing the partial bisimulation equivalence quotient in the vein of [13, 15] using the splitting technique of [24, 11]. The algorithm has improved complexity over previous work in minimization by simulation, while retaining comparable spatial requirements. For technical details and proofs we refer to the supporting technical report [4].

2 Process Algebra

In this section we define a basic sequential process algebra $\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B})$ with complete synchronization and a partial bisimilarity preorder. We also give a modal characterization of the preorder. The parameters of the process algebra are a finite set of actions $\mathcal{A}$ and a bisimulation set $B \subseteq \mathcal{A}$, which plays a role in the behavioral relation relation that will be revealed shortly. The nomenclature follows the approach of [2]. The partial bisimilarity preorder has been proposed in [28] as a relational characterization of controllability in a language-theoretic setting. It seamlessly caters for nondeterminism of the specifications as well. First, we deal with the non-recursive part of the process algebra and, afterwards, we extend it with guarded recursion.

2.1 Signature

The following definition gives the signature of the process terms.

Definition 2.1. The signature of the terms of the process algebra $\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B})$ is given by:

$$P ::= 0 \mid 1 \mid a.P \mid P + P \mid \ P|P,$$

where $a \in \mathcal{A}$. The set of (closed) process terms induced by P is denoted by $\mathcal{T}$.

The constant process 0 denotes inaction, i.e., it cannot execute any action and it can only deadlock. The constant process 1 denotes the option to successfully terminate. For each action $a \in \mathcal{A}$, the process corresponding to the term $a.p$ is capable of executing the action a and it continues behaving as p. The binary operator $\_ + \_$ denotes alternative composition. The process corresponding to the term $p + q$ makes a non-deterministic choice and behaves as p or as q. The binary operator $\_ \mid \_$ denotes synchronous parallel composition. The process $p \mid q$ synchronizes all actions of p and q and if no actions can be synchronized, it deadlocks.

2.2 Operational Semantics

We give structural operational semantics for each process term $p \in \mathcal{T}$. The semantics is given in terms of labeled graphs with successful termination, labeled graphs for short, modulo a behavioral equivalence. A labeled graph, defined by $G = (\mathcal{N}, \mathcal{L}, \downarrow, \longrightarrow)$, has a set of nodes $\mathcal{N}$, which are connected by transitions labeled by $\mathcal{L}$ and defined by the relation $\longrightarrow \subseteq \mathcal{N} \times \mathcal{L} \times \mathcal{N}$. Some nodes are marked by the predicate $\downarrow \subseteq \mathcal{N}$ as having the successful termination option. For a process term $p \in \mathcal{T}$ we have a labeled graph of the form $(\mathcal{T}, \mathcal{A}, \downarrow, \longrightarrow)$, where $\downarrow$ and $\longrightarrow$ are defined using by the structural operational rules given in Fig. 1. We will use infix notation and write $p\downarrow$ for $p \in \downarrow$ and $p \xrightarrow{a} p'$ for $(p, a, p') \in \longrightarrow$.

$$\textbf{1}\ \frac{}{1\downarrow} \qquad \textbf{2}\ \frac{p\downarrow}{p+q\downarrow} \qquad \textbf{3}\ \frac{q\downarrow}{p+q\downarrow} \qquad \textbf{4}\ \frac{p\downarrow,\ q\downarrow}{p \mid q\downarrow}$$

$$\textbf{5}\ \frac{}{a.p \xrightarrow{a} p} \qquad \textbf{6}\ \frac{p \xrightarrow{a} p'}{p+q \xrightarrow{a} p'} \qquad \textbf{7}\ \frac{q \xrightarrow{a} q'}{p+q \xrightarrow{a} q'} \qquad \textbf{8}\ \frac{p \xrightarrow{a} p',\ q \xrightarrow{a} q'}{p \mid q \xrightarrow{a} p' \mid q'}$$

Figure 1: Operational rules for the operators

We briefly comment on the operational rules. Rule 1 states that the constant process 1 enables successful termination. Rules 2 and 3 show that if one component of the alternative composition has a termination option, then the alternative composition has a termination option

as well. Rule 4 states that the synchronous parallel composition has a termination option only if both components have a termination option. Rule 5 states that action prefixes induce outgoing transitions with the same label. Rules 6 and 7 enable a nondeterministic choice between the alternatives of the parallel composition. Rule 8 states that in a synchronous parallel composition both components execute in lock-step always executing the same actions.

We use the predicates $p \xrightarrow{a}$ and $p \xrightarrow{a} \!\!\!\!\!/$ to denote that p has or does not have an outgoing transition labeled by a, respectively. Let $\pi = a_1 a_2 \ldots a_n \in \mathcal{A}^*$. By $p \xrightarrow{\pi} \!\!\!\!\!\rightarrow p'$, we denote that there exist $p_1, p_2, \ldots, p_{n-1} \in \mathcal{T}$ such that $p \xrightarrow{a_1} p_1 \xrightarrow{a_2} p_2 \xrightarrow{a_3} \ldots p_{n-1} \xrightarrow{a_n} p'$ for some trace (path) $\pi = a_1 a_2 \ldots a_n \in \mathcal{A}^*$. Standardly, by ε we denote the empty trace and by $\pi_1 \pi_2$ we denote the concatenation of the traces $\pi_1, \pi_2 \in \mathcal{A}^*$. By $\mathrm{T}(p)$ we denote the set of traces of p, i.e., $\mathrm{T}(p) = \{\pi \mid p \xrightarrow{\pi} \!\!\!\!\!\rightarrow\}$.

2.3 Partial Bisimilarity

In this section we revisit the notion of the partial bisimilarity preorder of [28] and we show that it is a precongruence for the given operations.

Definition 2.2. Let R be a relation on $\mathcal{T}$. Then R is a partial bisimulation with respect to the bisimulation set B if for all $p, q \in \mathcal{T}$ such that $(p, q) \in R$ the following holds:

1. if $p\downarrow$, then $q\downarrow$;
2. for all $p' \in \mathcal{T}$ and $a \in \mathcal{A}$ such that $p \xrightarrow{a} p'$, there exists $q' \in \mathcal{T}$ such that $q \xrightarrow{a} q'$ and $(p', q') \in R$; and
3. for all $q' \in \mathcal{T}$ and $b \in B$ such that $q \xrightarrow{b} q'$, there exists $p' \in \mathcal{T}$ such that $p \xrightarrow{b} p'$ and $(p', q') \in R$.

We say that the process term p is partially bisimilar to q with respect to the bisimulation set B, notation $p \preceq_B q$, if there exists a partial bisimulation R with respect to B such that $(p, q) \in R$. If $p \preceq_B q$ and $q \preceq_B p$, then we say that p and q are mutually partially bisimilar (with respect to B) and we write $p \leftrightarrow_B q$. When clear from the context, we will omit B.

It can be easily shown that partial bisimilarity is a preorder relation [28]. Also, it is not difficult to prove that mutual partial bisimilarity is an equivalence relation [28]. Note that if the bisimulation set B is empty, i.e., $B = \emptyset$, then the partial bisimilarity preorder coincides with the standard (strong) similarity preorder and the partial bisimilarity equivalence coincides with standard similarity equivalence [14, 12]. When $B = \mathcal{A}$, the partial bisimilarity preorder becomes strong bisimilarity provided that condition 1. is strengthened to "$p\downarrow$ if and only if $q\downarrow$", whereas mutual partial bisimilarity always turns into standard (strong) bisimilarity [14, 2].

We have the following property that characterizes the dependence on the bisimulation set B.

Theorem 2.3. *If $p \preceq_B q$, then $p \preceq_C q$ for every $C \subseteq B$.*

Proof. Straightforward from Definition 2.2 as condition 3. holds for every $b \in C$ as it holds for $b \in B$. □

The following property states that for two process terms to be mutually partially bisimilar with respect to B it is sufficient that partial bisimulation holds in one direction and simulation in the other.

Theorem 2.4. *If $p \preceq_B q$ and $q \preceq_\emptyset p$, then $p \leftrightarrow_B q$ for all $p, q \in \mathcal{T}$ and $B \subseteq \mathcal{A}$.*

Proof. Let R_1 be a partial bisimulation with respect to B such that $(p, q) \in R_1$ and let R_2 be a simulation such that $(q, p) \in R_2$. We will show that $R = R_1^{-1} \cap R_2$ is a partial bisimulation with respect to B. It is clear that $(q, p) \in R$. Suppose that $(r, s) \in R$. Then, $(r, s) \in R_1$ and $(s, r) \in R_2$.

Suppose that $s\downarrow$. Since $(s, r) \in R_2$, we have that $r\downarrow$. Suppose that there exist $s' \in \mathcal{T}$ and $a \in \mathcal{A}$ are such that $s \xrightarrow{a} s'$. Then, there exists r' such that $r \xrightarrow{a} r'$ and $(s', r') \in R_2$. As $r \xrightarrow{a} r'$ and $(r, s) \in R_1$, there exists s''' such that $s \xrightarrow{a} s'''$ and $(r', s''') \in R_1$. We repeat this process finitely many times coming to $\hat{r}$ and $\hat{s}$ such that $r \xrightarrow{a} \hat{r}$ and $s \xrightarrow{a} \hat{s}$ with $(\hat{r}, \hat{s}) \in R_1$ and $(\hat{s}, \hat{r}) \in R_2$ implying that $(\hat{s}, \hat{r}) \in R$. The case when $r \xrightarrow{b} r'$ for some $b \in B$ is analogous. □

The following theorem states that partial bisimilarity $\preceq_B$ is a precongruence with respect to the prefix, alternative composition, and synchronization operations.

Theorem 2.5. *Let $p, q \in \mathcal{T}$ and suppose $p \preceq_B q$ for some $B \subseteq \mathcal{A}$. Then:*

- *$a.p \preceq_B a.q$, for all $a \in \mathcal{A}$.*
- *$p + r \preceq_B q + r$ and $r + p \preceq_B r + q$, for all $r \in \mathcal{T}$.*
- *$p \mid r \preceq_B q \mid r$ and $r \mid p \preceq_B r \mid q$, for all $r \in \mathcal{T}$.*

Proof. Suppose $p \preceq q$. Then there exists a partial bisimulation relation R such that pRq as given by Definition 2.2. We define for each case a separate partial bisimulation relation R' based on R. We only show one of the symmetrical cases for the alternative and parallel composition, as the other holds by symmetry of the operational rules.

- Define $R' = \{(a.p, a.q)\} \cup R$. The terms $a.p$ and $a.q$ cannot terminate and have the outgoing transitions $a.p \xrightarrow{a} p$ and $a.q \xrightarrow{a} q$ with $(p, q) \in R$.
- Define $R' = \{(p+r, q+r)\} \cup \{(r, r) \mid r \in \mathcal{T}\} \cup R$. The relation $R'' = \{(r, r) \mid r \in \mathcal{T}\}$ is a partial bisimulation relation. The term $p + r$ terminates due to a termination options of p or due to a termination option of r. In the former situation $q + r$ terminates due to a termination option of q and in the latter due to a termination option of r. After the initial transition, the remaining terms are related by R'', if r is chosen, or by R, if p and q are chosen.
- Define $R' = \{(p \mid r, q \mid r) \mid (p, q) \in R,\ r \in \mathcal{T}\}$. The terms $p \mid r$ and $q \mid r$ either have termination options due to coinciding termination of p, q, and r or do not terminate. According to the operational semantics $p \mid r \xrightarrow{a}$ only if $p \xrightarrow{a}$ for some $a \in \mathcal{A}$. As (p, q) are related by R, it follows directly that R' is a partial bisimulation relation.

□

We have the following immediate corollary by taking into account Definition 2.2.

Corollary 2.6. *Partial bisimilarity $\leftrightarrow$ is a congruence for $\mathcal{T}$ with respect to the operators $a.\_$ for $a \in \mathcal{A}$, $\_ + \_$, and $\_ \mid \_$.*

Theorem 2.5 and Corollary 2.6 provide for substitution rules in the equational reasoning.

Now, we can build the standard term model [2] for the process algebra $\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B})$ by using partial bisimilarity as the underlying behavioral congruence.

Definition 2.7. The term algebra $\mathbb{P}(\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B}))$ is given by

$$\mathbb{P}(\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B})) = (\mathcal{T}, 0, 1, a.\_ \text{ for } a \in \mathcal{A}, \_ + \_, \_ \mid \_).$$

The term model of $\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B})$ is given by the quotient algebra $\mathbb{P}(\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B}))_{/\leftrightarrow}$.

2.4 Axiomatization

We give a sound and ground-complete axiomatization of the precongruence $\preceq$. When we write $p = q$ we mean that the axioms $p \leq q$ and $q \leq p$ are both included in the axiomatization. The axiomatization is given in Fig. 2.

$p + q = q + p$	A1	$(p + q) + r = p + (q + r)$	A2
$p + p = p$	A3	$p + 0 = p$	A6
$p \mid q = q \mid p$	S1	$0 \mid p = 0$	S2
$1 \mid 1 = 1$	S3	$1 \mid a.q = 0$	S4
$a.p \mid a.q = a.(p \mid q)$	S5	$a.p \mid b.q = 0, \quad \text{if } a \neq b$	S6
$(p + q) \mid r = p \mid r + q \mid r$	S7		
$p \leq p + 1$	P1	$q \leq a.p + q, \quad \text{if } a \notin B$	P2

Figure 2: Axiomatization of $\preceq$ over $\mathcal{T}$

We briefly comment on the axioms. We note that the numbering of the axioms follows [2]. Axioms A1 and A2 express commutativity and associativity of the alternative composition, respectively. Axiom A3 shows that the alternative composition is idempotent. Axiom A6 states that deadlock does not contribute to any behavior. Axiom S1 shows that the parallel composition is commutative. Deadlock cannot synchronize with any process as expressed by axiom S2. Axioms S3 and S4 state that the successful termination option persists only if it is enabled by both processes. Axiom S5 states that processes with the same prefix synchronize, whereas axiom S6 states that no synchronization is possible between processes with different prefixes. The distribution law of the synchronous parallel composition with respect to the alternative composition is stated by axiom S7. Axiom P1 enables elimination of the successful termination option for partially bisimulated terms. Axiom P2 enables elimination of terms that are prefixed by actions that do not have to be bisimulated.

We note that when the bisimulation set $B = \emptyset$ axiom P2 is valid for every possible prefix, effectively replacing axioms P1 and P2 with $q \leq_{\emptyset} p + q$. Thus, $\mathrm{BSP}_{|}(\mathcal{A}, \emptyset)$ reduces to the sound and ground-complete process theory for standard similarity preorder [14, 12]. When $B = \mathcal{A}$ axiom P2 becomes inapplicable, as there are no actions in $\emptyset = \mathcal{A} \setminus B$. Then, the remaining axioms minus axiom P1, which allows elimination of the successful termination option, form a sound and ground-complete process theory for standard bisimulation [2, 14] of a process algebra with action prefix, alternative composition, and synchronous parallel composition. The following theorem states the axiomatization is sound and ground-complete for the partial bisimilarity preorder.

Theorem 2.8. *The axioms of* $\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B})$ *given in Fig. 2 are sound and ground-complete for partial bisimilarity, i.e.,* $p \leq_B q$ *is derivable if and only if* $p \preceq_B q$.

Proof. Axioms A1, A2, A3, A6, S1, S2, S3, S4, S5, S6, and S7 make a sound and ground-complete axiomatization for strong bisimilarity, i.e., for $\leq_{\mathcal{A}}$ [2]. Thus, from now on, we can assume that $B \neq \mathcal{A}$.

The soundness of axioms P1 and P2 follows directly by application of the operational rules and Definition 2.2 for partial bisimilarity. It is sufficient to take $R = \{(p, p+1)\} \cup \{(p,p) \mid p \in \mathcal{T}\}$ and $R' = \{(q, a.p+q)\} \cup \{(q,q) \mid q \in \mathcal{T}\}$ as partial bisimulations between the terms for axiom P1 and P2, respectively. For axiom P1 it is clear that $p+1$ terminates if p terminates and they have the same outgoing labeled transitions. For axiom P2, if $q \xrightarrow{c} q'$ for some $q' \in \mathcal{T}$ and $c \in \mathcal{A}$, $a.p + q \xrightarrow{c} q'$ and $(q', q') \in R$. Vice versa, the outgoing transitions labeled by $b \in B$ of $a.p + q$ must originate from q as $a.p$ has only one outgoing transition labeled by $a \notin B$. Therefore, if $a.p + q \xrightarrow{b} q'$ for some $q' \in \mathcal{T}$ and $b \in B$, then $q \xrightarrow{b} q'$ and $(q', q') \in R$.

In order to show ground-completeness, we turn to normal forms as outlined, e.g., in [1, 2]. By using the axioms (for strong bisimilarity) every term $p \in \mathcal{T}$ can be rewritten as $p =_{\mathcal{A}} \sum_{i \in I} a_i.p_i[+1]$, where $a_i \in \mathcal{A}$ and $p_i \in \mathcal{T}$ for $i \in I$, and $[+1]$ denotes successful termination as an optional summand [2].

Now, suppose the normal forms of p and q are:

$$p =_{\mathcal{A}} \sum_{i \in I} a_i.p_i[+1] \qquad \text{and} \qquad q =_{\mathcal{A}} \sum_{j \in J} c_j.q_j[+1],$$

where $a_i, c_j \in \mathcal{A} \setminus B$ and $p_i, q_j \in \mathcal{T}$, for $i \in I$ and $j \in J$. We denote the normal forms of p and q by p' and q', respectively. From $p \leftrightarrow_{\mathcal{A}} p'$ and Theorem 2.3 it follows that $p \leftrightarrow_B p'$. Analogously, we have $q \leftrightarrow_B q'$, so we can conclude that $p' \preceq_B q'$ if and only if $p \preceq_B q$. We note that there are no idempotent summands in the normal forms. Now, the proof can be performed using induction on the total number of symbols, i.e., constants and action prefixes, of the terms.

The base cases are $p' =_B 0 \leq_B 0 =_B q'$ and $p' =_B 1 \leq_B 1 =_B q'$, which hold directly by using the substitution rules in an empty context, and $p' =_B 0 \leq_B 1 =_B q'$, which is obtained by applying $0 \leq_B 0 + 1 =_B 1$.

As $p' \preceq_B q'$, there exists a partial bisimulation R such that $(p', q') \in R$. It is clear that if p' contains the optional summand 1, then q' contains it as well. If q' comprises the summand 1 and p' does not contain it, then we use axiom P1 to eliminate it. Suppose that $p' \xrightarrow{a} p''$ for some $a \in \mathcal{A}$ and $p'' \in \mathcal{T}$. Then, according to the operational rules there exists a summand $a_k.p_k$ of p' for some $k \in I$ such that $a_k = a$ and $p_k = p''$. Analogously, by Definition 2.2 there exists a summand $c_\ell.q_\ell$ of q', such that $c_\ell = a$ and $(p_k, q_\ell) \in R$ for some $\ell \in J$. So, $p_k \preceq_B q_\ell$ and, hence, by the induction hypothesis, $p_k \leq_B q_\ell$. Thus, there exists $L \subseteq J$ such that for every $i \in I$ there exists $\ell \in L$ such that $a_i.p_i \leq_B c_\ell.q_\ell$. Vice versa, for every $j \in J$ such that $c_j \in B$ there exists $k \in I$ such that $a_k.p_k \leq_B c_j.q_j$.

Denote by $K = L \cup \{j \mid c_j \in B, j \in J\}$. Now, we split q' to $q' = q'' + q'''$ such that q'' contains the summands that are prefixed by an action in B or that have an index in L and q''' comprises the remaining summands, i.e., $q'' = \sum_{k \in K} c_k.q_k$ and $q''' = \sum_{\ell \in J \setminus K} c_\ell.q_\ell$. Note that p''' contains only summands prefixed by actions that are not in B. Now, we have that $p =_B p' \leq_B q''$. By applying Axiom P2 for the summands $c_\ell.q_\ell$ of q''' for $\ell \in J \setminus K$ we obtain $q'' \leq_B q'' + q''' =_B q' =_B q$, leading to $p \leq_B q$, which completes the proof. □

We conjecture that the partial bisimilarity equivalence does not admit a finite axiomatization when $\emptyset \subset B \subset \mathcal{A}$. To illustrate this, we consider the set of equations $E = \{a.b^n.0 + a.b^n.a.0 =_B a.b^n.a.0 \mid a \notin B, b \in B, n \in \mathbb{N}\}$, where $b^n.p$ is defined recursively as $b^0.p \triangleq p$ and $b^{n+1}.p \triangleq b.b^n.p$. It is not difficult to check that $a.b^n.0 + a.b^n.a.0 \leftrightarrow_B a.b^n.a.0$ for every $n \in \mathbb{N}$. However at depth greater than 1, we have $b.p + b.q \leftrightarrow_B b.q$, which holds only when $p \leftrightarrow_B q$, which is not the case for $p \triangleq b^n.0$ and $q \triangleq b^n.a.0$. This insinuates that E is a set of axioms.

In the literature the summand $a.b^n.0$ is also known as the 'little brother summand' of $a.b^n.a.0$. Little brother summands occur when dealing with similarity-like equivalences and have a very important role in their characterization [7, 13, 5]. Two similar terms that do not contain little brother summands are actually strongly bisimilar. This claim holds immediately for partial bisimilarity as well, having in mind Theorem 2.3.

Definition 2.9. Let $p =_B a.p' + a.p'' + p'''$ for some $a \in \mathcal{A}$ such that $a.p' \leq_B a.p''$ holds, but $a.p'' \leq_B a.p'$ does not hold. Then, we say that $a.p'$ is the little brother of $a.p''$.

The following properties show the nature of mutual partial bisimilarity.

Theorem 2.10. *Suppose that $p \leq q \leq r$. Then, the following equations hold:*

$$a.p + a.q = a.q \quad \textit{if } a \notin B \quad \text{P3} \qquad b.p + b.q + b.r = b.p + b.r \quad \textit{if } b \in B \quad \text{P4}.$$

Proof. We show that the equations are sound by showing the inequalities in both directions. For equation P3 we have that $a.p + a.q \leq a.p$ holds directly by axiom P2. For the other direction we calculate $a.p = a.p + a.p \leq a.p + a.q$ using axiom A3 and the premise, respectively. For equation P4 we have the following derivation, calculated by using axiom A3 and the premise, accordingly:

$$b.p + b.q + b.r \leq b.p + b.r + b.r = b.p + b.r$$
$$b.p + b.r = b.p + b.p + b.r \leq b.p + b.q + b.r,$$

which completes the proof. □

These equations show how to eliminate little brother summands. We note that idempotency of the alternative composition, given by axiom A3, is used in the situation when $p \leq q$ and $q \leq p$, even though we do not distinguish this in the conditions of the equations above. Equation P3 is actually a more general form of the known characteristic equation of the (strong) similarity equivalence stated in the form $a.(p + q) + a.q = a.(p + q)$ in [14]. As for strong similarity the prefix action does not play any role the axiom is always applicable. To establish partial bisimilarity when the little brothers are prefixed by an action in the bisimulation set B, the 'littlest' and the 'biggest' brother must be preserved. The equations given by Theorem 2.10 set the rules for elimination of little brothers when deriving the minimal mutual partially bisimilar quotient.

2.5 Modal Characterization

We give a modal characterization of the partial bisimilarity preorder in the vein of [14]. The following definition gives the partial bisimilarity formulas.

Definition 2.11. The partial bisimilarity modal formulas are given by F defined as follows:

$$N ::= \top \mid \neg\top \mid \neg 1 \mid \neg\langle a\rangle F \mid \neg(F \wedge F) \mid \langle b\rangle N$$
$$F ::= \top \mid \neg\top \mid 1 \mid \langle a\rangle F \mid F \wedge F \mid \neg N,$$

where $a \in \mathcal{A}$ and $b \in B$. The set of all partial bisimilarity modal formulas with respect to a bisimulation set B is denoted by $\mathcal{F}(B)$. The satisfaction relation $\models \subseteq \mathcal{T} \times \mathcal{F}(B)$ is defined recursively by:

- $p \models \top$ for all $p \in \mathcal{T}$;
- $p \models 1$ if and only if $p\downarrow$;
- $p \models \langle a \rangle f$ if there exists $p' \in \mathcal{T}$ such that $p \xrightarrow{a} p'$ and $p' \models f$;
- $p \models \neg f$ if not $p \models f$;
- $p \models f \wedge g$ if $p \models f$ and $p \models g$.

Note that formulas given by N are negations of formulas defined by F. It preserves that the negation, which characterizes bisimilar behavior, is enabled only for actions in the bisimulation set B. The partial bisimilarity modal formulas are a superset of the modal formulas for similarity and a subset of the ones for bisimilarity because negation is not present in the former and allowed for all formulas in the latter. When $B = \emptyset$, the modal formulas given by F minus the successful termination predicate 1 reduce to the ones for similarity [14], i.e., F reduces to $F ::= \top \mid \neg\top \mid \langle a \rangle F \mid F \wedge F$. When $B = \mathcal{A}$ and, again, provided that we ignore the termination predicate 1, F reduces to $F ::= \top \mid \langle a \rangle F \mid F \wedge F \mid \neg F$, i.e., the Hennessy-Milner formulas over $\mathcal{A}$ that identify bisimulation [14]. The following theorem shows that a process p is partially bisimilar to a process q if and only if all partial bisimilarity modal formulas that are satisfied by p are also satisfied by q.

Theorem 2.12. *Let $p, q \in \mathcal{T}$. Then, $p \preceq_B q$ if and only if for every $f \in \mathcal{F}(B)$ it holds that if $p \models f$ then $q \models f$.*

Proof. First we show the implication from left to right. Suppose that $p \preceq_B q$ and $p \models f$ for some $f \in \mathcal{F}(B)$. We will show that $q \models f$ by structural induction on f.

- Suppose $f \equiv 1$. According to Definition 2.11 $p\downarrow$. As $p \preceq_B q$, we have that $q\downarrow$. Thus, $q \models 1$.
- Suppose $f \equiv \langle a \rangle f'$. Then $p \xrightarrow{a} p'$ with $p' \models f'$. We have that $q \xrightarrow{a} q'$ and $p' \preceq_B q'$. By the hypothesis $q' \models f'$ and, thus, $q \models f$.
- Suppose $f \equiv \neg\langle b \rangle f'$. Then for all $p' \in \mathcal{T}$ such that $p \xrightarrow{b} p'$, it holds $p' \models \neg f'$. Note that $\neg f' \in \mathcal{F}(B)$. As $p \preceq_B q$, we have that $q \xrightarrow{b}$. Suppose that $q \xrightarrow{b} q'$. We will show that $q' \models \neg f'$ by contradiction. Suppose that $q' \models f'$. Then, there exists $p'' \in \mathcal{T}$ such that $p \xrightarrow{b} p''$. From above, $p'' \models \neg f'$, so by the hypothesis $q \models \neg f'$, which leads to contradiction. Thus, $q' \models \neg f'$ implying that $q \models f$.
- Suppose $f \equiv f' \wedge f''$. Then $p \models f'$ and $p \models f''$ implying that $q \models f'$ and $q \models f''$ by the induction hypothesis, so $q \models f$.

Next, we show the implication to left. Suppose that for every $f \in \mathcal{F}$ such that $p \models f$ it holds $q \models f$ as well. We will show that there exists a partial bisimulation by induction on the number of constants and action prefixes in p.

The base cases are: (1) If $p \equiv 0$, then $p \models \neg\langle b \rangle \top$ for all $b \in B$ are all non-trivial formulas satisfied by p leading to $q \models \neg\langle b \rangle \top$. So, $q \not\xrightarrow{b}$ implying $p \preceq_B q$. (2) If $p \equiv 1$, then we have (1) and $p \models 1$ implying that $q \models 1$. So, $p\downarrow$ implies that $q\downarrow$.

Now, suppose that $p\downarrow$. Then $p \models 1$, so $q \models 1$ as well, implying $q\downarrow$. Suppose that $p \xrightarrow{a} p'$ and $p' \models f'$. Then, $p \models \langle a \rangle f'$, so $q \models \langle a \rangle f'$, i.e., $q \xrightarrow{a} q'$ and $q' \models f'$. By the induction hypothesis we have that $p' \preceq_B q'$. Finally, suppose that $q \xrightarrow{b} q'$ and $q' \models f'$ for some $f' \in \mathcal{F}$. We will show that $p \xrightarrow{b} p'$ and $p' \preceq_B q'$ by contradiction. It must be that $p \xrightarrow{b}$ because in the opposite case $p \models \neg\langle b \rangle \top$ implying $q \models \neg\langle b \rangle \top$ which leads to contradiction. Suppose that $p \xrightarrow{b} p'$ and $p' \models \neg f'$. Note that it must be that $\neg f' \in \mathcal{F}$. Then, $p \models \neg\langle b \rangle f'$, implying that $q \models \neg\langle b \rangle f'$. So, we have that for all q' such that $q \xrightarrow{b} q'$ it holds that $q' \models \neg f'$ leading to contradiction. Thus, $p' \models f'$, so $p' \preceq_B q'$ implying that $p \preceq_B q$, which completes the proof. □

2.6 Recursion

We introduce recursion by means of guarded essentially finite state recursive specifications, which induce finite state transition systems [3] to obtain $\mathrm{BSP}_{|}(\mathcal{A}, \mathrm{B}, \mathcal{R})$, where $\mathcal{R}$ is the set of recursion variables. We restrict only to such specifications as every finite state transition system can be specified as a guarded essentially finite state recursive specification [3]. The restriction is given by forcing seriality of recursive variables [3], i.e., no free occurrence of a recursive variable is in the scope of the parallel composition, as well as requiring that they are guarded, i.e., every recursive variable is encapsulated by the action prefix operator. Processes given as solutions to the recursive specifications have the following signature:

$$\mu X.\{X = G \mid X \in R, R \subseteq \mathcal{R}\},$$

which is added to the existing signature of the process algebra, where

$$G ::= P \mid a.T \mid G + G, \qquad T ::= X \mid G \mid T + T,$$

and P is given by Definition 2.1. We will denote the set of guarded essentially finite state recursive specifications by $\mathcal{S}$.

Before we introduce the standard operational rules, we give a useful notation [2]. By t_X we will denote the term defining variable X. Also, we generalize $\mu X.S$, for $S \in \mathcal{S}$, to $\mu p.S$, for $p \in \mathcal{T}$ using the following inductive definition:

Definition 2.13. Define μpS, for $p \in \mathcal{T}$ and $S \in \mathcal{S}$, using structural induction, as follows:

$$\begin{aligned}
&\mu 0.S = 0\\
&\mu 1.S = 1\\
&\mu(a.q).S = a.(\mu q.S)\\
&\mu(q + r).S = \mu q.S + \mu r.S\\
&\mu(\mu X.S).S = \mu X.S
\end{aligned}$$

Note that in p all free occurrences of X are replaced by $\mu X.S$.

Now, the standard operational rules for solutions of recursive specifications can be stated as given in Fig. 3.

$$9\ \frac{\mu t_X.S\downarrow}{\mu X.S\downarrow} \qquad 10\ \frac{\mu t_X.S \xrightarrow{a} p}{\mu X.S \xrightarrow{a} p}$$

Figure 3: Operational rules for the solutions of the recursive specifications

The axioms in Fig. 4 capture the recursive definition and recursive specification principles,

which state that every recursive specification has a solution and that the guarded recursive specifications have at most one solution [3, 2].

$$\mu X.S \uplus \{Y = t_Y\} = \mu(\mu X.S).\{Y = t_Y\} \quad \text{if } X \neq Y \qquad \textbf{A1}$$

$$\mu X.\{X = t\} = \mu t.\{X = t\} \qquad \textbf{A2}$$

$$\text{if } t\{p/X\} \leq p \text{ then } \mu X.\{X = t\} \leq p \qquad \textbf{A3}$$

$$\text{if } p \leq t\{p/X\} \text{ then } p \leq \mu X.\{X = t\} \qquad \textbf{A4}$$

Figure 4: Axioms for manipulation with recursive specifications

Axiom A1 enables decomposition of the recursive specification to only one equation. This provides for head normal forms which contain only recursive specification of the form $\mu X.X = t$ [3]. Axiom A2 is the standard unfolding axiom, recall Definition 2.13 for the extended syntax. Axioms A3 and A4 are the folding axioms, which originate from [12], where it is shown that they hold for simulation. The combination of axioms A3 and A4 gives rise to the standard folding axiom

$$\text{if } t\{p/X\} = p \text{ then } \mu X.\{X = t\} = p.$$

Next, we develop a partitioning algorithm for computing the mutual partial bisimilarity quotient.

3 Controllability

We define controllability from a process algebraic perspective in terms of partial bisimilarity preorder. Standardly, we split $\mathcal{A}$ into a set of uncontrollable actions $\mathcal{U} \subseteq \mathcal{A}$, and a set of controllable actions $\mathcal{C} = \mathcal{A} \setminus \mathcal{U}$. The plant, the control requirements, and the supervisor are specified as process terms, relying on $\mathrm{BSP}_{|}(\mathcal{A}, \mathcal{U})$. Intuitively, outgoing uncontrollable transitions of the plant should be bisimilar to the ones of the supervised plant, so that the reachable uncontrollable part of the former is indistinguishable from the one of the latter. The outgoing controllable transitions of the supervised plant may only be simulated by the ones of the original plant, since some undesired controllable transitions are suppressed by the supervisor. We use $p \in \mathcal{T}$ to denote the plant, $r \in \mathcal{T}$ for the control requirements, and $s \in \mathcal{T}$ to denote the supervisor. Consequently, the supervised plant is given by $p \mid s$. First, we introduce the control problem and, afterwards, we characterize the existence of a deterministic supervisor.

Definition 3.1. Let $p \in \mathcal{T}$ be the plant and $r \in \mathcal{T}$ be the control requirements. The control problem is to find a supervisor $s \in \mathcal{T}$ such that $p \mid s \leq_{\mathcal{U}} p$ and $p \mid s \leq_{\emptyset} r$.

As expected, Definition 3.1 ensures that no uncontrollable actions have been disabled in the supervised plant, by including them in the bisimulation set. Moreover, it takes into account the nondeterministic behavior of the system. It suggests that the plant is modeled as is, whereas the control requirements are modeled as desired behavior, independent of the plant. This is in contrast with much work done in this area, where the aim is to satisfy a given desired controllable behavior. Still, we opt for an 'external' specification in process algebraic spirit, where, e.g., one wants to show that a given protocol behaves like a buffer when abstracted from the internal protocol communication. In this context, the control requirement

is the buffer, whereas the protocol is treated as a plant. Following this approach we only require that the supervised plant has a behavior that can be simulated by the control requirements. The setting described above is also a preparation for future work, where we intend to relax this condition in the vein of [29, 20], abstracting from irrelevant internal actions in the control requirements, an approach advocated from process algebraic perspective as well. Nonetheless, hiding in [29, 20] is performed in trace semantics, whereas abstraction should preserve branching behavior. Moreover, in [29, 20] the goal is to achieve bisimilarity and similarity with the control requirements, respectively, again insinuating that the control requirements are seen as the abstracted behavior of the supervised plant. The approach of [10] couples the requirements with the plant even more closely, requiring that they play the role of the supervisor as well. Note that if we assume that r represents the desired behavior of the supervised plant, then we require that $r \leq_{\mathcal{U}} p$, since $r \leq_{\emptyset} r$, as in the original setting of [27]. When p and r are deterministic that this amounts to standard language controllability [28].

As argued above, we choose bisimilarity as an appropriate behavioral equivalence that captures nondeterminism. Therefore, one expects that when we take the plant as a control requirement, the resulting controllability conditions $p \mid s \leq_{\mathcal{U}} p$ and $p \mid s \leq_{\emptyset} p$ will amount to bisimilarity. The conditions collapse to $p \mid s \leq_{\mathcal{U}} p$, as $p \mid s \leq_{\mathcal{U}} p$ implies $p \mid s \leq_{\emptyset} p$. Now, we can seek the largest possible supervised plant, i.e., $p \leq_{\mathcal{U}} p \mid s$, leading to $p \mid s =_{\mathcal{U}} p$. Note that the plant can have redundant behavior in the form of little brothers, which prevents a bisimilarity between p and $p \mid s$ to be established. Nevertheless, under the assumption that no little brothers are present, $p \mid s =_{\mathcal{U}} p$ implies $p \mid s =_{\mathcal{A}} p$, as shown in [5] for similarity equivalence, further justifying the choice of partial bisimilarity preorder.

According to Definition 3.1, the minimal possible supervisor is the initial uncontrollable reach of the plant, given by the topmost subterm of p comprising only uncontrollable prefixes. For example, the minimal supervisor of $p \triangleq \mu X.\{X = u.X + c.u.X + v.c.0\}$, assuming that $p =_{\mathcal{U}} r$, $u, v \in \mathcal{U}$, and $c \in \mathcal{C}$, is the process $s \triangleq \mu X.\{X = u.X + v.0\}$. According to Definition 3.1, every plant becomes controllable with respect to itself, i.e., every plant can accept itself as control requirement. This is a downside of the notion of state controllability, used in the nondeterministic setting of [10, 29, 20]. As an illustration, let $p =_{\mathcal{U}} r$ with $p \triangleq u.v.0 + u.w.0$, where $u, v, w \in \mathcal{U}$. Then, the plant is not state controllable with respect to itself, which is directly checked using the definition from the introduction. However, a non-restricting supervisor $s \triangleq \mu X.\{X = u.X + v.X + w.X\}$, which enables all transitions, always exists. Another supervisor is the determinized version of the control requirements, given by $s' \triangleq u.(v.0 + w.0)$.

As illustrated above, a usual suspect for a deterministic supervisor is the determinized version of a desired supervised behavior. First, we define a determinized version $\det(p)$ of a process $p \in \mathcal{T}$. By $\mathrm{Tar}(p \xrightarrow{a}) \triangleq \{p' \in \mathcal{T} \mid p \xrightarrow{a} p'\}$ denote all target processes that are reachable from $p \in \mathcal{T}$ by an outgoing transition labeled by $a \in \mathcal{A}$. The determinized version of p is defined as follows:

$$\textbf{11}\ \frac{p\downarrow}{\det(p)\downarrow} \qquad\qquad \textbf{12}\ \frac{p \xrightarrow{a}}{\det(p) \xrightarrow{a} \det(\sum_{p' \in \mathrm{Tar}(p \xrightarrow{a})} p')}$$

Rule 11 states that the original and determinized process have the same termination options. Rule 12 merges a nondeterministic choice over equally labeled transitions to a single transition which target is the alternative composition of all original target processes. It is not difficult to observe that $p \mid \det(p) =_B p$ for all $p \in \mathcal{T}$ and $B \subseteq \mathcal{A}$, as $\det(p)$ does not disable any transition of p.

Suppose that a desired behavior of the supervised plant is given by $q \in \mathcal{T}$ such that $q \leq_{\mathcal{U}} p$ and $q \leq_{\emptyset} r$. The control problem is to find a supervisor $s \in \mathcal{T}$, such that $p \mid s =_{\mathcal{U}} q$. A good candidate is $s \triangleq \det(q)$, since from $q \leq_{\mathcal{U}} p$ we have that $q \mid \det(q) \leq_{\mathcal{U}} p \mid \det(q)$, implying $q \leq_{\mathcal{U}} p \mid \det(q)$. Now, we need a characterization when $p \mid \det(q) \leq_{\mathcal{U}} p$, as it does not hold in general.

Theorem 3.2. *For all $p, q \in \mathcal{T}$, $p|\det(q) \leq_{\mathcal{U}} p$ if and only if $\det(p)|\det(q) \leq_{\mathcal{U}} \det(p)$.*

Proof. Suppose that $p \mid \det(q) \leq_{\mathcal{U}} p$. Then, there exists a partial bisimulation R such that $(p \mid \det(q), p) \in R$. By $p \xrightarrow{t}\!\!\!\!\rightarrow p'$ denote that there exist $p_1, p_2, \ldots, p_{n-1} \in \mathcal{T}$ such that $p \xrightarrow{a_1} p_1 \xrightarrow{a_2} p_2 \xrightarrow{a_3} \ldots p_{n-1} \xrightarrow{a_n} p'$ for some trace $t = a_1 a_2 \ldots a_n \in \mathcal{A}^*$ with $p \xrightarrow{\varepsilon}\!\!\!\!\rightarrow p$. By $\mathrm{Tar}(p \xrightarrow{t}\!\!\!\!\rightarrow) \triangleq \{p' \in \mathcal{T} \mid p \xrightarrow{t}\!\!\!\!\rightarrow p'\}$ denote the target processes reachable from p following a trace t. Now, define $R' = \{(\det(\sum_{p' \in \mathrm{Tar}(p \xrightarrow{t}\!\!\!\!\rightarrow)} p'), \det(\sum_{p' \in \mathrm{Tar}(p \xrightarrow{t}\!\!\!\!\rightarrow)} p')) \mid (p'' \mid q', p''') \in R$ and there exists $t \in \mathcal{A}^*$ such that $p \xrightarrow{t}\!\!\!\!\rightarrow p''$ and $p \xrightarrow{t}\!\!\!\!\rightarrow p'''\}$. We will show that R' is a partial bisimulation. Note that $\det(p) \xrightarrow{t}\!\!\!\!\rightarrow \det(\sum_{p' \in \mathrm{Tar}(p \xrightarrow{t}\!\!\!\!\rightarrow)} p')$ for every $t \in \mathcal{A}^*$. For $t = \varepsilon$, we have that $(\det(p) \mid \det(q), \det(p)) \in R'$. Suppose that $(r' \mid q', r')$ with $r' \triangleq \det(\sum_{p' \in \mathrm{Tar}(p \xrightarrow{t'}\!\!\!\!\rightarrow)} p')$ for some $t' \in \mathcal{A}^*$. If $r' \mid q'\!\downarrow$ then, $r'\!\downarrow$ and $q'\!\downarrow$. Suppose that $r' \mid q' \xrightarrow{a} r'' \mid q''$ for some $a \in \mathcal{A}$ and $r'', q'' \in \mathcal{T}$. Then, $r' \xrightarrow{a} r''$ and there exist $p'_1, q'_1 \in \mathcal{T}$ such that $p \xrightarrow{t}\!\!\!\!\rightarrow p'_1$ and $q \xrightarrow{t}\!\!\!\!\rightarrow q'_1$ with $p'_1 \xrightarrow{a} p'_2$ and $q'_1 \xrightarrow{a} q'_2$. Note that q'_1 is uniquely determined. Then, $(p'_1 \mid q'_1, p''_1) \in R$ for some $p''_1 \in \mathcal{T}$ such that $p \xrightarrow{t}\!\!\!\!\rightarrow p''_1$ and $p''_1 \xrightarrow{a} p''_2$ with $(p'_2 \mid q'_2, p''_2) \in R$, implying that $(r'' \mid q'', r'') \in R'$. The proof when $r' \xrightarrow{b} r''$ for some $b \in B$ and $r'' \in \mathcal{T}$ is analogous.

Suppose that $\det(p)|\det(q) \leq_{\mathcal{U}} \det(p)$. Then there exists a partial bisimulation relation R such that $(\det(p)|\det(q), \det(p)) \in R$. Define $R' = \{(p' \mid q', p') \mid p \xrightarrow{t}\!\!\!\!\rightarrow p', \det(q) \xrightarrow{t}\!\!\!\!\rightarrow q'$ and $(p'' \mid q', p'') \in R$ where $\det(p) \xrightarrow{t}\!\!\!\!\rightarrow p''$ for $t \in \mathcal{A}^*\}$. Then, R' is a partial bisimulation relating $p \mid \det(q)$ and p. The proof follows the same lines as above. □

Relying on Theorem 3.2 and Theorem 2.4, we characterize when desired behavior given by $q \in \mathcal{T}$ is controllable with respect to plant $p \in \mathcal{T}$ and control requirements $r \in \mathcal{T}$.

Definition 3.3. Process $q \in \mathcal{T}$ is controllable with respect to plant $p \in \mathcal{T}$ and control requirements $r \in \mathcal{T}$, a if $q \leq_{\mathcal{U}} p$, $p \mid \det(q) \leq_{\emptyset} r$, $\det(q) \leq_{\mathcal{U}} \det(p)$, and $p \mid \det(q) \leq_{\emptyset} q$.

The definition requires that (1) the plant partially bisimulated and the requirements simulate the behavior of the supervised plant, so that Definition 3.1 is satisfied, i.e., it is ensured that the supervised behavior satisfies the requirements and it is compatible with the plant on the uncontrollable events; (2) the deterministic behavior of the supervised plant, i.e., its language, is partially bisimulated by the plant, implying that the deterministic version of the desired behavior of the supervised plant can be used as a supervisor; and (3) the supervised behavior should simulate the supervised plant, implying that they are mutually partially bisimilar. We note that if equivalence is not desired, i.e., the supervised plant should only contain the desired behavior, then we can eliminate the third condition.

The following theorem states the existence of a supervisor.

Theorem 3.4. *If $q \in \mathcal{T}$ is controllable with respect to a plant $p \in \mathcal{T}$ and control requirements $r \in \mathcal{T}$, then $\det(q)$ is a supervisor for p with respect to r.*

Proof. From $\det(q) \leq_{\mathcal{U}} \det(p)$ and Theorem 3.2 we have that $p \mid \det(q) \leq_{\mathcal{U}} p$. Then, from $p \mid \det(q) \leq_{\emptyset} q$ and Theorem 2.4 we have that $p \mid \det(q) =_{\mathcal{U}} q$. Finally, from $q \leq_{\mathcal{U}} p$ and $q \leq_{\emptyset} r$ we have that $p \mid \det(q) \leq_{\mathcal{U}} p$ and $p \mid \det(q) \leq_{\emptyset} r$, which completes the proof. □

We note that the minimal deterministic supervisor of the plant p such that the supervised plant contains the behavior of q, i.e., $q \leq_{\mathcal{U}} p$, is $\det(q)$. So, for any other supervisor $s \in \mathcal{T}$ that satisfies the above relation, we must have that $q \leq_{\emptyset} s$ and $\det(p)|\det(s) \leq_{\mathcal{U}} \det(p)$.

We can also demand that the control requirements r are controllable, i.e., we wish that the desired behavior of the plant is the same as the control requirement. This amounts to $r \leq_{\mathcal{U}} p$, $\det(r) \leq_{\mathcal{U}} \det(p)$, and $p \mid \det(r) \leq_{\emptyset} r$. It is directly observed that the first requirements ensured compatibility of the control requirements with the plant, the second requirement is equivalent to language controllability, whereas the third requirement induces a refinement relation of the behavior of the supervised and the control requirements, respectively, comparable to the approaches of [23, 29, 20]. We note that for deterministic systems, we the first and the second condition coincide. The requirements can be efficiently checked using the algorithm presented in the following section. Finally, note that the plant p can be replaced by any p' such that $p' =_{\mathcal{U}} p$, providing for a minimization procedure that preserves controllability.

4 Partial Bisimulation Algorithm

We give a partitioning algorithm for computing the mutual partial bisimilarity quotient of a given labeled graph. With small adjustments the algorithm can be used to check whether two labeled graphs are partially bisimilar, as for the similarity equivalence [13]. The algorithm exploits the idea that partial bisimilarity equivalence can be presented as a partition pair, as it was done for the similarity equivalence in [13, 26]. The algorithm presented in [13] was mended in [15]. An extended version with proofs concerning the stability conditions for similarity can be found in [26]. We improve upon these works by employing the efficient splitting technique of [24, 11].

Let $G = (\mathcal{N}, \mathcal{L}, \downarrow, \longrightarrow)$ be a labeled graph. A set $\mathcal{P}$ is a *partition* over $\mathcal{N}$ if $\mathcal{P} \subset 2^{\mathcal{N}}$ such that $\bigcup_{P \in \mathcal{P}} P = \mathcal{N}$ and for all $P, Q \in \mathcal{P}$ if $P \cap Q \neq \emptyset$, then $P = Q$. A *partition pair* over G is a pair $(\mathcal{P}, \sqsubseteq)$ where $\mathcal{P}$ is a partition over $\mathcal{N}$ and the (little brother) relation $\sqsubseteq \subseteq \mathcal{P} \times \mathcal{P}$ is a partial order, i.e., a reflexive, antisymmetric, transitive relation. We note that our definition is stronger in the sense that we require $\sqsubseteq$ to be antisymmetric and transitive, opposed to only acyclic as originally defined in [13].

For all $P \in \mathcal{P}$, by $P\!\downarrow$ and $P\!\not\downarrow$ we denote that $p\!\downarrow$ and $p\!\not\downarrow$, respectively, for all $p \in P$. For $P' \in \mathcal{P}$ by $p \xrightarrow{a} P'$ we denote that there exists $p' \in P'$ such that $p \xrightarrow{a} p'$. We distinguish two types of (Galois) transitions between the partition classes [13, 16]: $P \xrightarrow{a}_{\exists} P'$, which denotes that there exists $p \in P$ such that $p \xrightarrow{a} P'$, and $P \xrightarrow{a}_{\forall} P'$, which denotes that for every $p \in P$ it holds that $p \xrightarrow{a} P'$. It is straightforward that $P \xrightarrow{a}_{\forall} P'$ implies $P \xrightarrow{a}_{\exists} P'$. Also, if $P \xrightarrow{a}_{\forall} P'$, then $Q \xrightarrow{a}_{\forall} P'$ for every $Q \subseteq P$. By $p \not\xrightarrow{a}_{\exists} P$ and $p \not\xrightarrow{a}_{\forall} P$ we denote that there are no transitions $p \xrightarrow{a}_{\exists} P$ and $p \not\xrightarrow{a}_{\forall} P$, respectively. The following definition gives the stability conditions for partial bisimilarity of a partition pair with respect to the termination predicate and the transition relation.

Definition 4.1. Let $G = (\mathcal{N}, \mathcal{L}, \downarrow, \longrightarrow)$ be a labeled graph. We say that a partition pair $(\mathcal{P}, \sqsubseteq)$ over G is stable with respect to B, $\downarrow$, and $\longrightarrow$ if the following conditions are fulfilled:

a. For all $P \in \mathcal{P}$ it holds that $P\downarrow$ or $P\not\downarrow$.

b. For all $P, Q \in \mathcal{P}$ such that $P \sqsubseteq Q$ if $P\downarrow$, then $Q\downarrow$.

c. For every $P, Q, P' \in \mathcal{P}$ and $a \in \mathcal{A}$ such that $P \sqsubseteq Q$ and $P \xrightarrow{a}_{\exists} P'$ there exists $Q' \in \mathcal{P}$ such that $P' \sqsubseteq Q'$ and $Q \xrightarrow{a}_{\forall} Q'$.

d. For every $P, Q, Q' \in \mathcal{P}$ and $b \in B$ such that $P \sqsubseteq Q$ and $Q \xrightarrow{b}_{\exists} Q'$ there exists a P such that $P \sqsubseteq Q$ and $P \xrightarrow{b}_{\forall} P'$.

Note that when $B = \mathcal{A}$, from $P \sqsubseteq Q$ we can straightforwardly deduce that $Q \sqsubseteq P$ by interchanging stability conditions *c* and *d*. Therefore, stability conditions *c* and *d* become: for every $P \in \mathcal{P}$ and $a \in \mathcal{A}$, if $P \xrightarrow{a}_{\exists} P'$, then $P \xrightarrow{a}_{\forall} P'$, which is the stability condition for the bisimulation equivalence [11]. When $B = \emptyset$, stability condition *d* is inapplicable, whereas stability condition *c* is the stability condition for the simulation preorder [13, 26].

Given a relation $R \in S \times T$, define $R^{-1} \in T \times S$ as $R^{-1} = \{(t, s) \mid (s, t) \in R\}$. If R is a preorder, then $R \cap R^{-1}$ is an equivalence relation. The following theorem shows that every partial bisimulation preorder induces a stable partition pair.

Theorem 4.2. *Let $G = (\mathcal{N}, \mathcal{A}, \downarrow, \longrightarrow)$ with $\mathcal{N} \subset \mathcal{T}$ and let R be a partial bisimulation preorder over $\mathcal{N}$ with respect to B. Let $\leftrightarrow \triangleq R \cap R^{-1}$. If $\mathcal{P} = \mathcal{T}/\!\leftrightarrow$ and for all $p, q \in \mathcal{N}$ it holds if $(p, q) \in R$, then $[p]_{\leftrightarrow} \sqsubseteq [q]_{\leftrightarrow}$, then the partition pair $(\mathcal{P}, \sqsubseteq)$ is stable with respect to B, $\downarrow$, and $\longrightarrow$.*

Proof. Let $P = [p]_{\leftrightarrow}$, $P' = [p']_{\leftrightarrow}$, $P'' = [p'']_{\leftrightarrow}$, $Q = [q]_{\leftrightarrow}$, $Q' = [q']_{\leftrightarrow}$, and $Q'' = [q'']_{\leftrightarrow}$ for $p, p', p'', q, q', q'' \in \mathcal{N}$. First, we show that $\sqsubseteq$ is a partial order. Reflexivity holds as for all $p' \in [p]_{\leftrightarrow}$ it holds that $(p, p') \in R$ implying $P \sqsubseteq P$. To show antisymmetry, suppose that $P \sqsubseteq Q$ and $Q \sqsubseteq P$. Then $(p, q) \in R$ and $(q, p) \in R$, implying $(q, p), (p, q) \in R^{-1}$ and $P = Q$. Finally, suppose that $P \sqsubseteq P'$ and $P' \sqsubseteq P''$. Then $(p, p'), (p', p'') \in R$. As R is a preorder, we have $(p, p'') \in R$ implying that $P \sqsubseteq P''$. So, $(\mathcal{P}, \sqsubseteq)$ is a partition pair.

Next, we show that the stability conditions of Definition 4.1 hold.

1. Suppose that $p\downarrow$. For every $p' \in [p]_{\leftrightarrow}$ it holds that $p \leftrightarrow p'$, so $p'\downarrow$ implying $P\downarrow$. Analogously for $p\not\downarrow$.

2. Let $P, Q \in \mathcal{P}$ be such that $P \sqsubseteq Q$. Now, if $P\downarrow$, then $p\downarrow$, which implies that $q\downarrow$ and also $Q\downarrow$.

3. Suppose that $P \sqsubseteq Q$ and $P \xrightarrow{a}_{\exists} P'$. Then, there exist $p \in P$ and $p' \in P'$ such that $p \xrightarrow{a} p'$. As $(p, q) \in R$ and $\sqsubseteq$ is a partial order, there exists $Q' \in \mathcal{P}$ such that $q \xrightarrow{a} q'$ and $(p', q') \in R$, and for all $q''' \in \mathcal{N}$ if $q \xrightarrow{a} q'''$ and $(p', q''') \in R$ then $Q''' \sqsubseteq Q'$ or Q''' and Q' are unrelated. Now, let $\bar{q} \in Q$. As $(q, \bar{q}) \in R$ there exists $\bar{q}' \in \bar{Q}'$ such that $\bar{q} \xrightarrow{a} \bar{q}'$ and $(q', \bar{q}') \in R$. Then $Q' \sqsubseteq \bar{Q}'$. As $(\bar{q}, q) \in R$ then exists $q'' \in Q''$ such that $q \xrightarrow{a} q''$ and $(\bar{q}', q'')$. Then $Q' \sqsubseteq \bar{Q}' \sqsubseteq Q''$ implying that $Q' = \bar{Q}' = Q''$. Thus, $Q \xrightarrow{a}_{\forall} Q'$.

4. Suppose that $P \sqsubseteq Q$ and $Q \xrightarrow{b}_{\exists} Q'$. The proof that there exists $P' \sqsubseteq Q'$ such that $P \xrightarrow{b}_{\forall} P'$ is analogous to 2.

□

Vice versa, every stable partition pair induces a partial bisimulation preorder.

Theorem 4.3. *Let $G = (\mathcal{N}, \mathcal{A}, \downarrow, \longrightarrow)$ with $\mathcal{N} \subset \mathcal{T}$ and let $(\mathcal{P}, \sqsubseteq)$ be a partition pair. Define $R = \{(p, q) \mid P \sqsubseteq Q, p \in P, q \in Q\}$. If $(\mathcal{P}, \sqsubseteq)$ is stable with respect to B, $\downarrow$, and $\longrightarrow$, then R is a partial bisimulation preorder.*

Proof. Let $P = [p]_{\leftrightarrow}$, $P' = [p']_{\leftrightarrow}$, $P'' = [p'']_{\leftrightarrow}$, $Q = [q]_{\leftrightarrow}$, $Q' = [q']_{\leftrightarrow}$, and $Q'' = [q'']_{\leftrightarrow}$ for $p, p', p'', q, q', q'' \in \mathcal{N}$. Suppose $(p, q) \in R$. In that case $P \sqsubseteq Q$. We will show that the stability conditions of Definition 2.2 hold for R.

1. If $p\downarrow$, then $P\downarrow$. So, $Q\downarrow$ implying $q\downarrow$.
2. Suppose $p \xrightarrow{a} p'$ for some $a \in \mathcal{A}$. Then, $P \xrightarrow{a}_{\exists} P'$ implying that there exists $Q' \in \mathcal{P}$ such that $Q \xrightarrow{a}_{\forall} Q'$. It follows that there exists q' such that $q \xrightarrow{a} q'$ and $(p', q') \in R$.
3. Suppose $q \xrightarrow{b} q'$ for some $b \in B$. Then, $Q \xrightarrow{b}_{\exists} Q'$ implying that there exists $P' \in \mathcal{P}$ such that $P \xrightarrow{b}_{\forall} P'$. It follows that there exists p' such that $p \xrightarrow{b} p'$ and $(p', q') \in R$.

□

Next, lets us define a partial order $\lhd$ on the partition pairs as follows.

Definition 4.4. Let $(\mathcal{P}, \sqsubseteq)$ and $(\mathcal{P}', \sqsubseteq')$ be partition pairs. We say that $(\mathcal{P}, \sqsubseteq)$ is finer than $(\mathcal{P}', \sqsubseteq')$, notation $(\mathcal{P}, \sqsubseteq) \lhd (\mathcal{P}', \sqsubseteq')$, if and only if for all P, Q such that $P \sqsubseteq Q$ there exist $P', Q' \in \mathcal{P}'$ such that $P \subseteq P'$, $Q \subseteq Q'$, and $P' \sqsubseteq' Q'$.

Now, it is not difficult to observe that finding the $\lhd$-maximal stable partition pair over a labeled graph G coincides with the problem of finding the coarsest partial bisimulation preorder over G. This is expressed by the following theorem.

Theorem 4.5. *Let $G = (\mathcal{N}, \mathcal{A}, \downarrow, \longrightarrow)$ with $\mathcal{N} \subset \mathcal{T}$. The $\lhd$-maximal partition pair $(\mathcal{P}, \sqsubseteq)$ stable with respect to B, $\downarrow$, and $\longrightarrow$ is induced by the partial bisimilarity preorder $\preceq_B$, i.e., $\mathcal{P} = \mathcal{N}/{\leftrightarrow_B}$ and $[p]_{\leftrightarrow_B} \sqsubseteq [q]_{\leftrightarrow_B}$ if and only if $p \preceq_B q$.*

Proof. By Theorem 4.2, the partition pair $(\mathcal{P}, \sqsubseteq)$ is stable with respect to B, $\downarrow$, and $\longrightarrow$. Suppose that $(\mathcal{P}', \sqsubseteq')$ is another partition pair that is stable with respect to B, $\downarrow$, and $\longrightarrow$. Then by Theorem 4.3 it induces a partial bisimulation preorder R' with respect to B. It is straightforward that $R' \subseteq \preceq_B$. We will show that $(\mathcal{P}', \sqsubseteq') \lhd (\mathcal{P}, \sqsubseteq)$. Suppose that $P' \sqsubseteq' Q'$ for some $P', Q' \in \mathcal{P}'$. Let $p' \in P'$ and $q' \in Q'$. Then, $(p', q') \in R'$, implying that $p' \preceq_B q'$. It follows that $[p']_{\leftrightarrow_B} \sqsubseteq [q']_{\leftrightarrow_B}$, which completes the proof. □

The algorithm iteratively refines the partition pair $(\mathcal{P}'_0, \sqsubseteq'_0) = (\{\mathcal{N}\}, \{(\mathcal{N}, \mathcal{N})\})$ over $G = (\mathcal{N}, \mathcal{A}, \downarrow, \longrightarrow)$ until it reaches the $\lhd$-maximal stable partition pair. We refine the partition by choosing *splitters*, which represent subsets of nodes that do not adhere to the stability conditions in combination with other nodes from the same class and, therefore, must be placed in a separate class. This induces a refinement of the partition, since a larger class, referred to as *parent*, is split into two or more classes. We manipulate with the splitters in the vein of [24, 11], which optimizes the computation of the refinements. For that reason, we need an initial set of splitters, which is computed according to the termination options and the outgoing labeled transitions of the states as follows.

Condition *a* of Definition 4.1 requires that all states in a class have or, alternatively, do not have termination options. Thus, we can immediately split $\mathcal{N}$ to P_0 and P_1 such that $P_0\downarrow$ and $P_1\not\downarrow$ with $P_0 \sqsubseteq P_1$. Note that at this point the relation $\sqsubseteq$ denotes the potential of P_0 being the little brother of P_1. We are certain that $P_1 \sqsubseteq P_0$ is not possible for any refinement of the partition $\{P_0, P_1\}$ because of stability condition *b* of Definition 4.1. Moreover, as any further refinement will actually refine $(\mathcal{P}_{00}, \sqsubseteq_{00}) \triangleq (\{P_0, P_1\}, \{(P_0, P_0), (P_0, P_1), (P_1, P_1)\}$ stability conditions *a* and *b* will always be satisfied, so we no longer have to take them into consideration during refinement. Next, let *the set of outgoing labels* $\mathrm{OL}(p) \triangleq \{a \in \mathcal{A} \mid p \xrightarrow{a}\}$ denote the set of labels of all outgoing transitions of the node $p \in \mathcal{N}$. We refine $(\mathcal{P}_{00}, \sqsubseteq_{00})$ as follows.

Suppose that the partition pair $(\mathcal{P}_0, \sqsubseteq_0)$ such that $(\mathcal{P}_0, \sqsubseteq_0) \lhd (\mathcal{P}_{00}, \sqsubseteq_{00})$ is the desired partition pair of splitters. The partition $\mathcal{P}_0$ is defined as for every $P \in \mathcal{P}_0$, $p, q \in P$ if and only if $\mathrm{OL}(p) = \mathrm{OL}(q)$, since if p and q do not have the same sets of outgoing labels, then they cannot be mutually partially bisimilar. This induces the sets $\mathrm{OL}(P) \triangleq \mathrm{OL}(p)$ with $p \in P$. We define the little brother relation by looking into the sets of outgoing labels. Recall that partially bisimilar terms must have the same sets of outgoing labels that are also in the bisimulation set. The set of remaining outgoing labels of the little brother, which transitions only have to be simulated, is a subset of the corresponding set of outgoing labels of the other term. So, we put $P \sqsubseteq_0 Q$ if and only if (1) $\mathrm{OL}(P) \setminus B \subseteq \mathrm{OL}(Q) \setminus B$, and (2) $\mathrm{OL}(P) \cap B = \mathrm{OL}(Q) \cap B$. As $(\mathcal{P}_0, \sqsubseteq_0) \lhd (\mathcal{P}_{00}, \sqsubseteq_{00})$, we have that if $P \in P_1$ then $Q \in P_1$. The initial little brother relation $\sqsubseteq_0$ satisfies the conditions of Definition 4.1, i.e., it is a partial order.

It is easily observed that the stability conditions are not necessarily satisfied for $(\mathcal{P}_0, \sqsubseteq_0)$. However, if we consider $(\mathcal{P}_0, \sqsubseteq_0)$ with respect to the partition pair $(\mathcal{P}'_0, \sqsubseteq'_0)$ we see that the stability conditions are satisfied. For example, for all $P, Q \in \mathcal{P}_0$, $P' \in \mathcal{P}'_0$, and $a \in \mathcal{A}$ it holds that if $P \xrightarrow{a} P'$, then there exists $Q' \in \mathcal{P}'_0$ such that $P' \sqsubseteq Q'$ and $Q \xrightarrow{a} Q'$. It is clear that $(\mathcal{P}_0, \sqsubseteq_0) \lhd (\mathcal{P}'_0, \sqsubseteq'_0)$. Now, the idea behind the partitioning algorithm is to iteratively refine $(\mathcal{P}'_0, \sqsubseteq'_0)$ to $(\mathcal{P}'_n, \sqsubseteq'_n)$ and $(\mathcal{P}_0, \sqsubseteq_0)$ to $(\mathcal{P}_n, \sqsubseteq_n)$ for some $n \in \mathbb{N}$. For all $0 \leq i \leq n$ we have that $(\mathcal{P}_i, \sqsubseteq_i) \lhd (\mathcal{P}'_i, \sqsubseteq'_i)$ and the stability conditions are satisfied for $(\mathcal{P}_i, \sqsubseteq_i)$ with respect to $(\mathcal{P}'_i, \sqsubseteq'_i)$. Moreover, $(\mathcal{P}_n, \sqsubseteq_n)$ and $(\mathcal{P}'_n, \sqsubseteq'_n)$ is a fix point of the algorithm, i.e., $(\mathcal{P}_n, \sqsubseteq_n) = (\mathcal{P}'_n, \sqsubseteq'_n)$. The refinement of $(\mathcal{P}', \sqsubseteq')$ and $(\mathcal{P}, \sqsubseteq)$ employs the splitters, which are initially formed of classes, which nodes cannot be combined as they contravene the stability conditions. The $\lhd$-maximality is achieved by showing that if a splitter contains nodes from two different classes, then it is unstable, e.g., as it was done for the initial classes above, implying that the resulting partition is the coarsest possible refinement.

Now, suppose that the partition pair $(\mathcal{P}, \sqsubseteq)$ has $(\mathcal{P}', \sqsubseteq')$ as parent with $(\mathcal{P}, \sqsubseteq) \lhd (\mathcal{P}', \sqsubseteq')$. For convenience, we rewrite the stability conditions *c* and *d* of Definition 4.1 for the partition pair $(\mathcal{P}, \sqsubseteq)$ with respect to $(\mathcal{P}', \sqsubseteq')$ and the bisimulation set B:

1. For all $P \in \mathcal{P}$, $a \in \mathcal{A}$, and $P' \in \mathcal{P}'$ such that $P \xrightarrow{a}_{\exists} P'$ there exists $Q' \in \mathcal{P}'$ such that $P' \sqsubseteq' Q'$ and $P \xrightarrow{a}_{\forall} Q'$.

2. For all $P, Q \in \mathcal{P}$, $a \in \mathcal{A}$, and $P' \in \mathcal{P}'$ such that $P \sqsubseteq Q$ and $P \xrightarrow{a}_{\forall} P'$ there exists $Q' \in \mathcal{P}'$ such that $P' \sqsubseteq' Q'$ and $Q \xrightarrow{a}_{\forall} Q'$.

3. For all $P \in \mathcal{P}$, $Q' \in \mathcal{P}'$, and $b \in B$ such that $P \xrightarrow{b}_{\exists} Q'$ there exists $P' \in \mathcal{P}'$ such that $P' \sqsubseteq' Q'$ and $P \xrightarrow{b}_{\forall} P'$.

4. For all $P, Q \in \mathcal{P}$, $Q' \in \mathcal{P}'$, and $b \in B$ such that $P \sqsubseteq Q$ and $Q \xrightarrow{b}_{\forall} Q'$ there exists $P' \in \mathcal{P}'$ such that $P' \sqsubseteq' Q'$ and $P \xrightarrow{b}_{\forall} P'$.

It is not difficult to observe that stability conditions 1 and 2 replace stability condition *c* of Definition 4.1, where as stability conditions 3 and 4 replace stability condition *d* of Definition 4.1. They are equivalent when $(\mathcal{P}, \sqsubseteq) = (\mathcal{P}', \sqsubseteq')$. From now on, we refer to the stability conditions above instead of the ones in Definition 4.1. The form of stability conditions as given above is useful as stability conditions 1 and 3 can be used to split the classes, whereas stability conditions 2 and 4 can be used to adjust the little brother relation.

We proceed in the vein of [24, 11], and we define the function $\mathrm{cnt}\colon \mathcal{N} \times \mathcal{A} \times 2^{\mathcal{N}} \to \mathbb{N}$ to optimize the splitting process, where $\mathrm{cnt}(p \xrightarrow{a} P') \geq 0$ denotes the number of transitions labeled by $a \in \mathcal{A}$ from the node $p \in \mathcal{N}$ to the parent $P' \in \mathcal{P}'$. Suppose that we refine the partition pair $(\mathcal{P}', \sqsubseteq')$ that is a parent of $(\mathcal{P}, \sqsubseteq) \lhd (\mathcal{P}', \sqsubseteq')$, where $P' \in \mathcal{P}'$ is such that $S' \subset P'$. The refinement step splits P' to S' and $P' \setminus S'$ and subsequently splits every class in $\mathcal{P}$ with respect to the splitter S' in order to satisfy the stability conditions. If one knows $\mathrm{cnt}(p \xrightarrow{a} P')$ and $\mathrm{cnt}(p \xrightarrow{a} S')$ has been computed for every node $p \in \mathcal{N}$ and action $a \in \mathcal{A}$, then this information can be used to update the function cnt for $P' \setminus S'$ and deduce the following:

0. If $\mathrm{cnt}(p \xrightarrow{a} P') = \mathrm{cnt}(p \xrightarrow{a} S') = 0$, then $p \not\xrightarrow{a} S'$, $p \not\xrightarrow{a} P' \setminus S'$, and $\mathrm{cnt}(p \xrightarrow{a} P' \setminus S') = 0$.
1. If $\mathrm{cnt}(p \xrightarrow{a} P') > 0$ and $\mathrm{cnt}(p \xrightarrow{a} S') = 0$, then $p \not\xrightarrow{a} S'$, $p \xrightarrow{a} P' \setminus S'$, and $\mathrm{cnt}(p \xrightarrow{a} P' \setminus S') = \mathrm{cnt}(p \xrightarrow{a} P')$.
2. If $\mathrm{cnt}(p \xrightarrow{a} P') = \mathrm{cnt}(p \xrightarrow{a} S') > 0$, then $p \xrightarrow{a} S'$, $p \not\xrightarrow{a} P' \setminus S'$, and $\mathrm{cnt}(p \xrightarrow{a} P' \setminus S') = 0$.
3. If $\mathrm{cnt}(p \xrightarrow{a} P') > 0$, $\mathrm{cnt}(p \xrightarrow{a} S') > 0$, and $\mathrm{cnt}(p \xrightarrow{a} S') \neq \mathrm{cnt}(p \xrightarrow{a} P')$, then $p \xrightarrow{a} S'$, $p \xrightarrow{a} P' \setminus S'$, and $\mathrm{cnt}(p \xrightarrow{a} P' \setminus S') = \mathrm{cnt}(p \xrightarrow{a} P') - \mathrm{cnt}(p \xrightarrow{a} S')$.

The advantage of this approach is that in order to decide where the node belongs after the splitting of P one has to compute $\mathrm{cnt}(p \xrightarrow{a} S')$ only for the nodes of S', which are less than the nodes of P', optimally as close to half as possible. Thus, in our initialization step we also need to compute $\mathrm{cnt}(p \xrightarrow{a} \mathcal{N})$ for every node $p \in \mathcal{N}$ and action $a \in \mathcal{A}$. This finishes the initialization phase, which delivers an initial partition pair $(\mathcal{P}_0, \sqsubseteq_0)$ with a parent $(\mathcal{P}'_0, \sqsubseteq'_0) = (\mathcal{N}, \{\mathcal{N}, \mathcal{N}\})$, and a corresponding cnt function. The time complexity of this phase is $\mathcal{O}(|\mathcal{N}||\mathcal{A}|)$ [5].

We proceed with the description of the refinement steps of the algorithm. Suppose that we want to split $P' \in \mathcal{P}'$ to $S', P' \setminus S' \in \mathcal{P}'$ for some $\emptyset \subset S' \subset P'$. The splitters $P' \setminus S'$ and S' should be chosen consistently, i.e., $S' \cap Q = Q$ or $S' \cap Q = \emptyset$ for all $Q \in P'$, $|S'| \leq \frac{|P'|}{2}$, and both $S' \sqsubseteq' P' \setminus S'$ and $P' \setminus S' \sqsubseteq' S'$ should not hold. One can always choose the $\sqsubseteq$-minimal or $\sqsubseteq$-maximal class of $\mathcal{P}$ with parent P' as such a splitter. We have to ensure that the stability conditions hold for $P' \setminus S'$ and S'. If $P \not\xrightarrow{a}_{\exists} P'$, then $P \not\xrightarrow{a}_{\exists} P' \setminus S'$ and $P \not\xrightarrow{a}_{\exists} S'$, which trivially satisfies the stability conditions. Suppose that $P \xrightarrow{a}_{\exists} P'$ for some $P \in \mathcal{P}$ and $a \in \mathcal{A}$. Then there exists $Q' \in \mathcal{P}'$ such that $P' \sqsubseteq' Q'$ and $P \xrightarrow{a}_{\forall} Q'$. If, in addition, $a \in B$, then there exists $Q'' \in \mathcal{P}'$ such that $Q'' \sqsubseteq P'$ and $P \xrightarrow{a}_{\forall} Q''$. Note that the above relations do not necessarily hold for $P' \setminus S'$ and S'. We distinguish the following situations. If $P \xrightarrow{a}_{\exists} P' \setminus S'$, then there must exist a $Q' \in \mathcal{P}'$, such that $P' \setminus S' \sqsubseteq' Q'$, $P' \setminus S' \neq Q'$, and $P \xrightarrow{a}_{\forall} Q'$, so that P is stable, dependent on $P' \setminus S' \sqsubseteq' Q'$. Note that we treat the case when $P \xrightarrow{a}_{\forall} P' \setminus S'$ below, as a splitting of P to P_1 or P_3. If $a \in B$, then additionally there must exist a $Q'' \in \mathcal{P}'$, such that $Q'' \sqsubseteq' P' \setminus S'$, $Q'' \neq P' \setminus S'$, and $P \xrightarrow{a}_{\forall} Q''$, implying that stability of P in this situation depends on $Q'' \sqsubseteq' P' \setminus S'$. Again, we treat the case when $P \xrightarrow{a}_{\forall} P' \setminus S'$ below, as

a splitting of P to P_1 or P_3. When $P \xrightarrow{a}_{\exists} S'$ we have a similar situation that the splitting of P depends on some $T' \in \mathcal{P}'$ such that $S' \sqsubseteq' T'$, $S' \neq T'$, and $P \xrightarrow{a}_{\forall} T'$ and, moreover, if $a \in B$, there must also exist $T'' \in \mathcal{P}'$ such that $T'' \sqsubseteq' S'$, $T'' \neq S'$, and $P \xrightarrow{a}_{\forall} T''$. When $P \xrightarrow{a}_{\forall} S'$, we treat it as splitting of P to P_2 or P_3 below. In the opposite case, P has to be split as discussed below.

Suppose that some $P \in \mathcal{P}$ is not split due to a dependence on $P' \sqsubseteq' Q'$ for some $P', Q' \in \mathcal{P}'$. Suppose that $P \xrightarrow{a}_{\exists} P'$ and $P \xrightarrow{a}_{\forall} Q'$ for some $a \in \mathcal{A}$ in order to satisfy stability condition 2. If we have to split P' to $P' \setminus S'$ and S' for some $S' \in \mathcal{P}'$, the dependence on $P' \sqsubseteq' Q'$ is no longer necessary, as we will consider $P \xrightarrow{a}_{\exists} P' \setminus S'$ and $P \xrightarrow{a}_{\exists} S'$. However, if we have to split Q' to $Q' \setminus T'$ and T' for some $T' \in \mathcal{P}'$, then the stability condition 2 for P no longer holds. In this case, we have to find some $Q'' \in \mathcal{P}'$ such that $Q' \sqsubseteq' Q''$ and $P \xrightarrow{a}_{\forall} Q''$. If such Q'' does not exist, then we have to split P accordingly. We have an analogous discussion when $Q' \sqsubseteq' P'$.

Suppose that $P \in \mathcal{P}$ does not conform to the stability conditions for some action $a \in \mathcal{A}$. We can potentially split P into four disjoint subsets P_0, P_1, P_2, and P_3 such that $P = P_0 \uplus P_1 \uplus P_2 \uplus P_3$, where $\uplus$ denotes disjoint union. The classes P_0, P_1, P_2, and P_3 contain nodes that satisfy the splitting conditions 0–3 from above, respectively, i.e., $P_0 \xrightarrow{a}\!\!\!\!\!/\;_{\exists} P' \setminus S'$, $P_0 \xrightarrow{a}\!\!\!\!\!/\;_{\exists} S'$, $P_1 \xrightarrow{a}_{\forall} P' \setminus S'$, $P_1 \xrightarrow{a}\!\!\!\!\!/\;_{\exists} S'$, $P_2 \xrightarrow{a}\!\!\!\!\!/\;_{\exists} P' \setminus S'$, $P_2 \xrightarrow{a}_{\forall} S'$, $P_3 \xrightarrow{a}_{\forall} P' \setminus S'$, and $P_3 \xrightarrow{a}_{\forall} S'$. Note that $P \xrightarrow{a}_{\forall} P'$ if and only if $P_0 = \emptyset$.

Suppose that $a \notin B$ and suppose that $P' \setminus S'$ and S' are not related. Then we have to split P to P_0, P_1, P_2, and P_3 as the stability condition 1 is not satisfied for P. Note that it is directly checked that stability condition 1 is satisfied for P_0, P_1, P_2, and P_3. Also, it should be clear that if we mix states from the classes, we obtain a subset of P, say Q, such that $Q \xrightarrow{a}_{\exists} P' \setminus S'$ and $Q \xrightarrow{a}\!\!\!\!\!/\;_{\forall} P' \setminus S'$, or $Q \xrightarrow{a}_{\exists} S'$ and $Q \xrightarrow{a}\!\!\!\!\!/\;_{\forall} S'$. We show that the stability condition 1 does not hold for Q. Suppose that stability condition 1 holds and take $Q \xrightarrow{a}_{\exists} S'$ and $Q \xrightarrow{a}\!\!\!\!\!/\;_{\forall} S'$. The discussion when $Q \xrightarrow{a}_{\exists} P' \setminus S'$ is analogous. If Q is stable, then there exists $T' \in \mathcal{P}'$ such that $S' \sqsubseteq' T'$ and $Q \xrightarrow{a}_{\forall} T'$. Note that $T' \neq S'$ and $T' \neq P' \setminus S'$. As $Q \subseteq P$, we have that $P \xrightarrow{a}_{\exists} T'$. As the stability condition for P holds for parents that are not $P' \setminus S'$ or S', it follows that there exists $Q' \in \mathcal{P}$ such that $T' \sqsubseteq' Q'$, implying $S' \sqsubseteq' Q'$, and $P \xrightarrow{a}_{\forall} Q'$. However, having in mind that a similar results is obtained when $Q \xrightarrow{a}_{\exists} P' \setminus S'$, we obtain a contradiction with the assumption that P is not stable with respect to $P' \setminus S'$ and S'. The classes are related as follows according to the stability condition 2: $P_0 \sqsubseteq P_0$, $P_0 \sqsubseteq P_1$, $P_0 \sqsubseteq P_2$, $P_0 \sqsubseteq P_3$, $P_1 \sqsubseteq P_1$, $P_1 \sqsubseteq P_3$, $P_2 \sqsubseteq P_2$, $P_2 \sqsubseteq P_3$, and $P_3 \sqsubseteq P_3$. When some of the classes are empty, the corresponding pairs are omitted. Now, suppose that $P' \setminus S' \sqsubseteq S'$. Recall that $P_2 \xrightarrow{a}\!\!\!\!\!/\;_{\exists} P' \setminus S'$ and $P_2 \xrightarrow{a}_{\forall} S'$, and $P_3 \xrightarrow{a}_{\forall} P' \setminus S'$ and $P_3 \xrightarrow{a}_{\forall} S'$. Consequently, we have that $P_2 \sqsubseteq P_3$ and $P_3 \sqsubseteq P_2$, i.e., P should be split to P_0, P_1, and $P_2 \uplus P_3$, dependent on $P' \setminus S' \sqsubseteq S'$, with $P_0 \sqsubseteq P_0$, $P_0 \sqsubseteq P_1$, $P_0 \sqsubseteq P_2 \uplus P_3$, $P_1 \sqsubseteq P_1$, $P_1 \sqsubseteq P_2 \uplus P_3$, and $P_2 \uplus P_3 \sqsubseteq P_2 \uplus P_3$. Finally, suppose that $a \in B$. Then, P is split to P_0, P_1, P_2, and P_3, where the classes are unrelated with each other, since the stability condition 4 is not satisfied for the pairs from above, i.e., it holds only that $P_0 \sqsubseteq P_0$, $P_1 \sqsubseteq P_1$, $P_2 \sqsubseteq P_2$, and $P_3 \sqsubseteq P_3$.

After the the refinement of $\mathcal{P}$ due to the splitting of P' to $P' \setminus S'$ and S', we have to adjust the little brother relation between the newly obtained classes. Suppose that $P \sqsubseteq Q$ held before the splitting of $P' \in \mathcal{P}'$ to $P' \setminus S'$ and S' and P was not split. Then for all $Q' \in \mathcal{P}'$ such that $P' \setminus S' \neq Q'$ and $S' \neq Q'$, if $P \xrightarrow{a}_{\forall} Q'$ for some $a \in \mathcal{A}$, then $Q \xrightarrow{a}_{\forall} Q''$ for some $Q'' \in \mathcal{P}'$ such that $Q' \sqsubseteq Q''$, implying that Q is not split as well. Similarly, for $a \in B$. When $P \xrightarrow{a}_{\forall} P' \setminus S'$ or $P \xrightarrow{a}_{\forall} S'$, then this is treated as P being split to P_1 or P_3, or P_2 or P_3, respectively. If P was split, and Q remained intact, then $P \sqsubseteq Q$ holds still if $P \xrightarrow{a}_{\exists} P' \setminus S'$

implies that $Q \xrightarrow{a}_{\exists} P' \setminus S'$ and if $P \xrightarrow{a}_{\exists} S'$ implies that $Q \xrightarrow{a}_{\exists} S'$. Suppose that for some $P, Q \in \mathcal{P}$, it held that $P \sqsubseteq Q$, before the splitting of P to P_0, P_1, P_2, and P_3, and Q to Q_0, Q_1, Q_2, and Q_3 with respect to action $a \in \mathcal{A}$. It should be clear that it is only possible that $P_i \sqsubseteq Q_j$ for some $i, j \in \{0, 1, 2, 3\}$. Following the reasoning from above, provided that $a \notin B$, and $P' \setminus S'$ and S' are unrelated, we will have the following pairs: $P_0 \sqsubseteq Q_0$, $P_0 \sqsubseteq Q_1$, $P_0 \sqsubseteq Q_2$, $P_0 \sqsubseteq Q_3$, $P_1 \sqsubseteq Q_1$, $P_1 \sqsubseteq Q_3$, $P_2 \sqsubseteq Q_2$, $P_2 \sqsubseteq Q_3$, and $P_3 \sqsubseteq Q_3$, for which the stability condition 2 is satisfied. To show this, it is sufficient to show that $P_i \sqsubseteq Q_i$ for $i \in \{0, 1, 2, 3\}$. As the stability condition 2 holds for P and Q, we have that if $P \xrightarrow{a}_{\forall} P''$ for some $P'' \in \mathcal{P}'$, then $Q \xrightarrow{a}_{\forall} Q'$ for some $Q' \in \mathcal{P}'$ such that $P'' \sqsubseteq' Q'$. As $P_i \subseteq P$ and $Q_i \subseteq Q$ for all $i \in \{0, 1, 2, 3\}$, we have that this is satisfied for all $P'' \neq P' \setminus S'$ and $P'' \neq S'$. Moreover, P_i and Q_i have the same $\xrightarrow{a}_{\forall}$ transitions to $P' \setminus S'$ and S', in which case the stability condition 2 is clearly satisfied. When $P' \setminus S' \sqsubseteq S'$, then we have $P_2 \uplus P_3 \sqsubseteq Q_2 \uplus Q_3$ as expected. When $a \in B$, then we have only that $P_i \sqsubseteq Q_i$ for all $i \in \{0, 1, 2, 3\}$. Again if some of the classes are empty, then the corresponding pairs are omitted. This finishes the splitting phase, which has complexity of $\mathcal{O}(| \longrightarrow | \log |\mathcal{N}|)$ for the splitting of P and $\mathcal{O}(|\mathcal{P}|^3)$ for updating the little brother relation. The former is well known [11, 5], whereas the latter is obtained by observing that there are $|\mathcal{P}|$ refinements in total, and for each the updating costs $\mathcal{O}(|\mathcal{P}|^2)$ [13, 26].

Note that after the splitting, the stability of $(\mathcal{P}, \sqsubseteq)$ with respect to the parent partition pair $(\mathcal{P}', \sqsubseteq')$ is dependent on the little brother relation $\sqsubseteq'$ between the parents. In return, recall that $\sqsubseteq'$ is dependent on the relation between the classes of $\mathcal{P}$, since for all $P', Q' \in \mathcal{P}'$, it holds that $P' \sqsubseteq' Q'$ if and only if there exist $P, Q \in \mathcal{P}$ with $P \in P'$, $Q \in Q'$, and $P \sqsubseteq Q$. So, after the refinement of the classes and the update of the little brother relation, we have to validate that all dependencies still hold. If they do not, then we need to split the classes, which depend on them accordingly, and repeat this procedure until no dependencies are conflicted. Suppose that $P \in \mathcal{P}$ has not been split due to a dependence of $P' \sqsubseteq' Q'$ for some $P', Q' \in \mathcal{P}'$, and $P \sqsubseteq Q$ for some $Q \in \mathcal{P}$. Then, either $P \xrightarrow{a}_{\exists} P'$ and $P \xrightarrow{a}_{\forall} Q'$, or $P \xrightarrow{a}_{\exists} Q'$ and $P \xrightarrow{a}_{\forall} P'$. Recall that if P is not split, then Q is not split as well, due to a dependence $P' \sqsubseteq' Q''$ or $Q'' \sqsubseteq' P'$, respectively, for some $Q'' \in \mathcal{P}'$ with $Q' \sqsubseteq' Q''$ or $Q'' \sqsubseteq Q'$, respectively. If there exists $T' \in \mathcal{P}'$ such that $P' \sqsubseteq' T'$ or $T' \sqsubseteq Q'$, respectively, with $P \xrightarrow{a}_{\forall} T'$, then P should not be split, and the dependence has to change. Since $P \sqsubseteq Q$, Q will not be split as well, for the same reasons as in the discussion above. In the opposite case, P needs to be split, implying that Q possibly has to be split as well, and in this case the little brother relation $\sqsubseteq$ is updated as above. The cost of the validation is again $\mathcal{O}(|\mathcal{P}|^3)$ [15].

Finally, when the stable partition pair $(\mathcal{P}, \sqsubseteq)$ of the labeled graph $G = (\mathcal{N}, \mathcal{L}, \downarrow, \longrightarrow)$ is reached, we have to construct the quotient transition system. For that purpose we use the properties of the partial bisimilarity equivalence of Theorem 2.10. The quotient is the label graph $H = (\mathcal{P}, \mathcal{L}, \downarrow, \longrightarrow')$, where the set of states are the classes of the partition, the set of labels is the same as for the original graph, the termination predicate is defined per class as above, and the transitions relation is defined as follows. For all $P, Q \in \mathcal{P}$ and $a \in \mathcal{A} \setminus B$ we put $P \xrightarrow{a}{}' Q$ if and only if there does not exist $Q' \in \mathcal{P}$ such that $Q \sqsubseteq Q'$ and $P \xrightarrow{a}_{\forall} Q'$. For all $P, Q \in \mathcal{P}$ and $b \in B$ we put $P \xrightarrow{b}{}' Q$ if and only if there does not exist $Q' \in \mathcal{P}$ such that $Q \sqsubseteq Q'$ and $P \xrightarrow{b}_{\forall} Q'$ or $Q' \sqsubseteq Q$ and $P \xrightarrow{b}_{\forall} Q'$. In other words, if the action transition has to be bisimulated, then we have to retain the littlest and the biggest brother, whereas if the action transitions needs to be only simulated, it is sufficient to retain the biggest brother, as in [13, 15]. The cost of the final step is $\mathcal{O}(|\mathcal{P}|^3)$. The spatial requirements are $\mathcal{O}(| \longrightarrow |) \log |\mathcal{N}|$ for the labeled graph and $\mathcal{O}(|\mathcal{A}||\mathcal{P}|^2)$ for the little brother relation and the dependencies, as there can be at most $|\mathcal{A}||\mathcal{P}|$ dependencies per class. The algorithm has time complexity of $\mathcal{O}((|\mathcal{A}| + \log |\mathcal{N}|)| \longrightarrow | + |\mathcal{P}|^3)$, which is better than $\mathcal{O}(| \longrightarrow ||\mathcal{P}|^2)$ for similarity [15]. The spatial requirements are comparable with $\mathcal{O}(|\mathcal{A}||\mathcal{P}|^2)$, needed for the little brother relation and the dependencies, compared to $\mathcal{O}(|\mathcal{P}|^2)$ of [15] needed for the little brother relation.

We summarize the algorithm for minimization by mutual partial bisimulation as follows.

Input: Labeled graph $G = (\mathcal{N}, \mathcal{L}, \downarrow, \longrightarrow)$

Output: Labeled graph $H = (\mathcal{P}, \mathcal{L}, \downarrow, \longrightarrow')$

1. Set $(\mathcal{P}', \sqsubseteq') := (\{\mathcal{N}\}, \{(\mathcal{N}, \mathcal{N})\})$. Compute $(\mathcal{P}, \sqsubseteq)$ such that for all $P, Q \in \mathcal{P}$ it holds that (1) $P\downarrow$ or $P\not\downarrow$; (2) if $P\downarrow$, then $Q\downarrow$; and (3) if $\mathrm{OL}(P) \setminus B \subseteq \mathrm{OL}(Q) \setminus B$ and $\mathrm{OL}(P) \cap B = \mathrm{OL}(Q) \cap B$, then $P \sqsubseteq Q$. For all $p \in \mathcal{N}$ compute $\mathrm{cnt}(p \xrightarrow{a} \mathcal{N})$.
2. Choose a splitter S' for parent $P' \in \mathcal{P}'$ as described above, and update $\sqsubseteq'$ for $P' \setminus S'$ and S'. If no such splitter exists, go to step 7.
3. For all $Q' \in \mathcal{P}$ such that $Q' \sqsubseteq' P'$ or $P' \sqsubseteq' Q'$ and the splitting of $P \in \mathcal{P}$ depends on this relation, i.e., $P \xrightarrow{a}_\exists Q'$ and $P \xrightarrow{a}_\forall P'$, check whether there exists $P'' \in \mathcal{P}'$ such that $Q' \sqsubseteq P''$ or $P'' \sqsubseteq Q'$, respectively, and $P \xrightarrow{a}_\forall P''$. If such P'' does exists, then transfer the dependence of splitting P to $Q' \sqsubseteq P''$ or $P'' \sqsubseteq Q'$, respectively. In the opposite case, split P accordingly.
4. For all $P \in \mathcal{P}$ and $a \in \mathcal{A}$ such that $P \xrightarrow{a}_\exists P'$, compute $\mathrm{cnt}(p \xrightarrow{a} S')$ for all $p \in P$ and, otherwise, set $\mathrm{cnt}(p \xrightarrow{a} P' \setminus S') := 0$ and $\mathrm{cnt}(p \xrightarrow{a} S') := 0$.
5. For all $a \in \mathcal{A}$ do the following:
   (a) For all $P \in \mathcal{P}$ such that $P \xrightarrow{a}_\exists P'$ do the following:
      i. If $P \xrightarrow{a}_\exists P' \setminus S'$, then check whether there exists $Q' \in \mathcal{P}'$ such that $P' \setminus S' \sqsubseteq' Q'$, $P' \setminus S' \neq Q'$, and $P \xrightarrow{a}_\forall Q'$. If $a \notin B$, then P should not be split, dependent on $P' \setminus S' \sqsubseteq' Q'$. If $a \in B$, then there should also exist $Q'' \in \mathcal{P}'$ such that $Q'' \sqsubseteq' P' \setminus S'$, $Q'' \neq P' \setminus S'$, and $P \xrightarrow{a}_\forall Q''$, making the splitting of P dependent on $Q'' \neq P' \setminus S'$ as well.
      ii. If $P \xrightarrow{a}_\exists S'$, then check whether there exists $T' \in \mathcal{P}'$ such that $S' \sqsubseteq' T'$, $S' \neq T'$, and $P \xrightarrow{a}_\forall T'$. If $a \notin B$, then P should not be split, dependent on $P' \setminus S' \sqsubseteq' Q'$. If $a \in B$, then there should also exist $T'' \in \mathcal{P}'$ such that $T'' \sqsubseteq' S'$, $T'' \neq S'$, and $P \xrightarrow{a}_\forall T''$, making the splitting of P dependent on $T'' \sqsubseteq' S'$ as well.
      iii. If the conditions in 5(a)i and 5(a)ii are not satisfied, then compute P_0, P_1, P_2, and P_3 as described above. If $P' \setminus S' \not\sqsubseteq S'$, then split P to P_0, P_1, P_2, and P_3 and, otherwise, split P to P_0, P_1, $P_2 \uplus P_3$ and update $\sqsubseteq$ as described above.
   (b) Update the little brother relation for $P \sqsubseteq Q$ as described above.
   (c) For all $P', Q' \in \mathcal{P}$ such that $P' \sqsubseteq' Q'$, if there do not exist $P \in P'$ and $Q \in Q'$ such that $P \sqsubseteq Q$, delete the relation between P' and Q'. For all $P \in \mathcal{P}$ that were dependent on $P' \sqsubseteq' Q'$ do the following. If P was not split due to dependence on $P' \sqsubseteq' Q'$, such that $P \xrightarrow{a}_\exists P'$ and $P \xrightarrow{a}_\forall Q'$, or $P \xrightarrow{a}_\exists Q'$ and $P \xrightarrow{a}_\forall P'$, check if there exists $Q'' \in \mathcal{P}'$ such that $P' \sqsubseteq' Q''$ or $Q'' \sqsubseteq Q'$, respectively, with $P \xrightarrow{a}_\forall Q''$. If such Q'' exists, replace the dependency on $P' \sqsubseteq' Q'$ with $P' \sqsubseteq' Q''$ or $Q'' \sqsubseteq P'$, respectively. On the contrary, split P.
   (d) Update the little brother relation $\sqsubseteq$ as described above. If there are any changes go back to step 5c to validate the dependencies.
6. If $(\mathcal{P}, \sqsubseteq) \neq (\mathcal{P}', \sqsubseteq')$ go to step 2.
7. Construct $H = (\mathcal{P}, \mathcal{L}, \downarrow, \longrightarrow')$ as described above.

In order to check the dependencies efficiently, we introduce a *topological order* of the classes, given a partition pair $(\mathcal{P}, \sqsubseteq)$ [13, 15]. We say that $\leq \subseteq 2^N$ is a topological order over $\mathcal{P}$ induced by $\sqsubseteq$ if for all $P, Q \in \mathcal{P}$ it holds that $P \leq Q$ if and only if $Q \not\sqsubseteq P$. Thus, if $P \leq Q$, then either $P \sqsubseteq Q$ or P and Q are unrelated. We note that the topological order in general is not uniquely defined. Also, we can represent the topological order as a list $P_1, P_2, \ldots, P_n$, for some $n \in \mathbb{N}$, where $P_i \leq P_j$ for $1 \leq i \leq j \leq n$. We have the following property that provides for efficient updating of the topological order. Let $(\mathcal{P}_1, \sqsubseteq_1)$ be a partition pair and let $\leq_1 = P_1, P_2, \ldots, P_n$, for some $n \in \mathbb{N}$, be a topological order over $\mathcal{P}_1$ induced by $\sqsubseteq_1$. Suppose that $P_i \in \mathcal{P}_1$ is split to $Q_1, Q_2 \in \mathcal{P}_2$, i.e., $\mathcal{P}_2 = \mathcal{P}_1 \setminus \{P_i\} \cup \{Q_1, Q_2\}$ such that $(\mathcal{P}_2, \sqsubseteq_2) \lhd (\mathcal{P}_1, \sqsubseteq_1)$. Suppose that $Q_1 \sqsubseteq_2 Q_2$ or Q_1 and $_2$ are unrelated. Then, $\leq_2$ given by $\leq_2 = P_1, \ldots, P_{i-1}, Q_1, Q_2, P_{i+1}, \ldots, P_n$ is a topological order over $\mathcal{P}_2$ induced by $\sqsubseteq_2$. To show this, recall that from $(\mathcal{P}_2, \sqsubseteq_2) \lhd (\mathcal{P}_1, \sqsubseteq_1)$, we have for all $P'', Q'' \in \mathcal{P}_2$ such that $P'' \sqsubseteq_2 Q''$, there exist $P', Q' \in \mathcal{P}_1$ such that $P'' \subseteq P'$, $Q'' \subseteq Q'$, and $P' \sqsubseteq_q Q'$. So, if $P' \leq_1 Q'$ then $P'' \leq_2 Q''$ or P'' and Q'' are unrelated for all $P'', Q'' \in \mathcal{P}_2$ such that $P'' \neq Q_1$ and $Q'' \neq Q_2$. Since $Q_1 \sqsubseteq_2 Q_2$ or Q_1 and Q_2 are unrelated, we have that $\leq_2$ is a topological order. This property enables us to update the topological order by replacing each class with the results of the splitting without having to re-compute it in every iteration, as it is done in [13, 15]. As a result, the classes whose nodes belong to the same parent are neighboring with respect to the topological order. Moreover, the topological order gives us a procedure for searching for a little or a big brother of a given class. All little brothers of a given class are topologically sorted in descendent to the left, and all the big brothers are topologically sorted ascendent to the right.

5 Concluding Remarks

We successfully employed the partial bisimilarity preorder to define controllability of nondeterministic processes. The definition of controllability is finer than existing notions in the literature and it reduces to language controllability for deterministic systems. To support this investigation we characterized the notion of partial bisimulation by means of sound and complete axiomatization and modal logic. We also developed a partitioning algorithm for minimization by partial bisimilarity that improves previous work and which is also suitable as a minimization procedure that preserves controllability.

As future work, we aim look into nonblocking supervisory control, as well as develop synthesis algorithms, and apply them to case studies. Another interesting topic is modular control, where the supervisor is split on synchronizing components. We also intend to investigate extensions with time and abstraction.

Bibliography

[1] L. Aceto, W. J. Fokkink, A. Ingolfsdottir, and B. Luttik. *Processes, Terms and Cycles: Steps on the Road to Infinity*, volume 3838 of *Lecture Notes in Computer Science*, chapter Finite Equational Bases in Process Algebra: Results and Open Questions, pages 338–367. Springer, 2005.

[2] J. C. M. Baeten, T. Basten, and M. A. Reniers. *Process Algebra: Equational Theories of Communicating Processes*, volume 50 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2010.

[3] J. C. M. Baeten and M. Bravetti. A ground-complete axiomatisation of finite-state processes in a generic process algebra. *Mathematical Structures in Computer Science*, 18(06):1057–1089, 2008.

[4] J. C. M. Baeten, D. A. van Beek, B. Luttik, J. Markovski, and J. E. Rooda. Partial bisimulation. SE Report 10-04, Eindhoven University of Technology, 2010.

[5] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[6] G. Barrett and S. Lafortune. Bisimulation, the supervisory control problem and strong model matching for finite state machines. *Discrete Event Dynamic Systems*, 8(4):377–429, 1998.

[7] D. Bustan and O. Grumberg. Simulation-based minimization. *ACM Transactions on Compututational Logic*, 4(2):181–206, 2003.

[8] C. Cassandras and S. Lafortune. *Introduction to discrete event systems*. Kluwer Academic Publishers, 2004.

[9] R. Eshuis and M. M. Fokkinga. Comparing refinements for failure and bisimulation semantics. *Fundamentae Informaticae*, 52(4):297–321, 2002.

[10] M. Fabian and B. Lennartson. On non-deterministic supervisory control. *Proceedings of the 35th IEEE Decision and Control*, 2:2213–2218, 1996.

[11] J.-C. Fernandez. An implementation of an efficient algorithm for bisimulation equivalence. *Science of Computer Programming*, 13(2-3):219–236, 1990.

[12] U. Frendrup and J. N. Jensen. A complete axiomatization of simulation for regular CCS expressions. BRICKS Report 01-26, University of Aarhus, Denmark, 2001.

[13] R. Gentilini, C. Piazza, and A. Policriti. From bisimulation to simulation: Coarsest partition problems. *Journal of Automated Reasoning*, 31(1):73–103, 2003.

[14] R. J. van Glabbeek. The linear time–branching time spectrum I. *Handbook of Process Algebra*, pages 3–99, 2001.

[15] R. J. van Glabbeek and B. Ploeger. Correcting a space-efficient simulation algorithm. In *Proceedings of CAV*, volume 5123 of *Lecture Notes in Computer Science*, pages 517–529. Springer, 2008.

[16] M. R. Henzinger, T. A. Henzinger, and P. W. Kopke. Computing simulations on finite and infinite graphs. In *IEEE Symposium on Foundations of Computer Science*, pages 453–462. IEEE Computer Society Press, 1996.

[17] M. Heymann and F. Lin. Discrete-event control of nondeterministic systems. *IEEE Transactions on automatic control*, 43(1):3–17, 1998.

[18] M. Heymann and G. Meyer. Algebra of discrete event processes. Technical Report NASA 102848, NASA Ames Research Center, 1991.

[19] R. Kumar and M. A. Shayman. Nonblocking supervisory control of nondeterministic systems via prioritized synchronization. *IEEE Transactions on automatic control*, 41(8):1160–1175, 1996.

[20] R. Kumar and C. Zhou. Control of nondeterministic discrete event systems for simulation equivalence. *IEEE Transactions on Automation Science and Engineering*, 4(3):340–349, 2007.

[21] K. G. Larsen. Modal specifications. In *Automatic Verification Methods for Finite State Systems*, volume 407 of *LNCS*, pages 232–246. Springer, 1990.

[22] K. G. Larsen and L. Xinxin. Equation solving using modal transitions systems. In *Proceedings of LICS*, pages 108–117. IEEE, 1990.

[23] A. Overkamp. Supervisory control using failure semantics and partial specifications. *IEEE Transactions on automatic control*, 42(4):498–510, 1997.

[24] R. Paige and R. E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.

[25] S.-J. Park and J.-T. Lim. Nonblocking supervisory control of nondeterministic systems based on multiple deterministic model approach. *IEICE Transactions on Information and Systems*, E83-D(5):1177–1180, 2000.

[26] B. Ploeger. *Improved Verification Methods for Concurrent Systems*. PhD thesis, Eindhoven University of Technology, 2009.

[27] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal on Control and Optimization*, 25(1):206–230, 1987.

[28] J. J. M. M. Rutten. Coalgebra, concurrency, and control. SEN Report R-9921, Center for Mathematics and Computer Science, Amsterdam, The Netherlands, 1999.

[29] C. Zhou, R. Kumar, and S. Jiang. Control of nondeterministic discrete-event systems for bisimulation equivalence. *IEEE Transactions on Automatic Control*, 51(5):754–765, 2006.