Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Bisimilarity on Basic Parallel Processes

Petr Jančar

Dept of Computer Science, Faculty of Science, Palacký University, Olomouc, Czechia

ARTICLE INFO

Article history:
Received 2 July 2021
Received in revised form 25 October 2021
Accepted 30 November 2021
Available online xxxx

Keywords:
Verification
Concurrency
Equivalence checking
Bisimulation equivalence
Basic parallel processes

ABSTRACT

The main aim of this paper is to give a simple transparent proof of the result showing that the problem of bisimulation equivalence on the class of Basic Parallel Processes (BPP), denoted by BPP-Bisim, can be decided by a polynomial-space algorithm. This result (by the author) has been previously only presented in a conference version, in a rather technical form that is not easily readable and verifiable. Since the result has clarified the complexity of the problem BPP-Bisim, namely its PSPACE-completeness (using the lower bound by Srba), and the problem deals with a fundamental behavioural equivalence and with one of the simplest models that naturally extend finite-state systems into infinite-state ones, it seems appropriate to have a transparent version of the proof.

1. Introduction

The main aim of this paper is to give a simple transparent proof of the result showing that the problem of bisimulation equivalence on the class of Basic Parallel Processes (BPP), denoted by BPP-Bisim, can be decided by a polynomial-space algorithm. This result (by the author) has been only presented in a conference version, namely in [1], in a rather technical form that is not easily readable and verifiable (as the author has also learned from other researchers). The result has thus clarified the complexity of the problem BPP-Bisim, namely its PSPACE-completeness, since its PSPACE-hardness was known due to Srba [2] (with the journal version [3]).

Since bisimulation equivalence is a fundamental behavioural equivalence in concurrency theory, having also natural connections with logics used in computer science (we can recall, e.g., [4] as one of the classical books), and Basic Parallel Processes constitute probably the simplest model that extends finite-state systems with a parallel-composition operation resulting in infinite-state systems (as discussed below), it seems appropriate to have also a reviewed journal version of the former conference presentation, which is now thoroughly rewritten.

We recall that bisimilarity (which is another name for bisimulation equivalence) is decidable in polynomial time on finite-state systems that can be presented as finite labelled transitions systems, or nondeterministic finite automata; the classical algorithms are given in [5,6]. We can also recall that bisimilarity is PTIME-complete for finite-state systems [7]. Regarding infinite-state systems, we can name [8] as a seminal paper showing the decidability of bisimilarity for one such class, namely for the class of normed BPA (Basic Process Algebra) processes. For further results in this area we can refer, e.g., to the surveys [9] and [10]; the latter has an updated online version.

The class BPP (in this paper meaning Basic Parallel Processes, *not* the complexity class with the same acronym) is, in fact, a parallel counterpart of the "sequential" class BPA. We can imagine that in a nondeterministic finite automaton, with the state set P (deliberately denoted by P to anticipate a representation by a Petri net with the set P of places) we

E-mail address: petr.jancar@upol.cz.

https://doi.org/10.1016/j.tcs.2021.11.027
0304-3975/

replace standard transitions $p \xrightarrow{a} p'$ with transitions $p \xrightarrow{a} M$ where M is generally a finite multiset of states in P; hence a "program thread" in state p can perform an action a by which it spawns a (maybe empty) collection of new threads in the prescribed states that run further in parallel (with no mutual communication). There are no accepting/rejecting states but we study the problem BPP-Bisim, i.e. the question if two multisets M_1, M_2 (also called configurations, or markings in the respective Petri nets) exhibit the same behaviour, in the sense that they are bisimilar, which is denoted by $M_1 \sim M_2$. (We recall that $M_1 \sim M_2$ iff each move $M_1 \xrightarrow{a} M_1'$ can be matched with a move $M_2 \xrightarrow{a} M_2'$ with the same action a so that $M_1' \sim M_2'$; moreover, $\sim$ is symmetric.)

The decidability of BPP-Bisim was shown by Christensen, Hirshfeld, and Moller [11], without any complexity upper bound. The problem Normed-BPP-Bisim, for the normed BPP, where from each multiset M we can reach the empty multiset $\mathbf{0}$, turned out to be substantially easier: it was shown to be in PTIME [12] (which was in [13] even extended to a class of normed BPP where a synchronization of parallel threads is allowed).

As already discussed, the proof presented here establishes that BPP-Bisim is in PSPACE, and is thus PSPACE-complete by using [3]. When the presented polynomial-space algorithm is restricted to the instances of Normed-BPP-Bisim, then it works in polynomial time; this yields a proof of the PTIME-membership that substantially differs from the proof in [12].[1]

The crux of the presented algorithm is that a certain semilinear (i.e. Presburger-arithmetic) representation of bisimulation equivalence on a BPP system, with the state set P, can be constructed effectively, and the membership of a given pair (M_1, M_2) in the represented set can be checked in polynomial space. In more detail, this check amounts to verify if $\mathrm{NORM}_Q(M_1) = \mathrm{NORM}_Q(M_2)$ for certain (maybe exponentially many) "important" sets $Q \subseteq \mathrm{P}$, where $\mathrm{NORM}_Q(M)$ is the *norm of a configuration* M w.r.t. the set Q, i.e., the length of a shortest path from M to a multiset whose intersection with Q is empty (the value $\mathrm{NORM}_Q(M)$ is infinite if no such multiset is reachable from M).

All this is informally explained, and then rigorously proved, in Section 3, after the preliminaries given in Section 2. Section 4 looks at the used approach from a more abstract viewpoint, of the "distance-to-disabling" functions on general labelled transition systems. In Section 5 we then briefly discuss related research on BPP and BPA, including the open problems regarding the equivalences called weak bisimilarity and branching bisimilarity, which are relevant in the presence of silent transitions. In particular, we also discuss a relation to the interesting notion of a certain *semantic norm* of configurations introduced by Yuxi Fu in [16], which initiated further research especially on the class BPA.

2. Notation, basic notions, and some standard facts

The aim of this section is to give a rigorous definition of the problem of bisimilarity on basic parallel processes (BPP), together with the used notation and some standard facts.

2.1. Numbers (with ω), multisets, partitions of sets

By $\mathbb{Z}, \mathbb{N}, \mathbb{N}_+$ we denote the sets of integers, nonnegative integers, and positive integers, respectively. We use ω just as a symbol denoting an infinite amount, and put $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$ and $\mathbb{N}_{\omega,-1} = \mathbb{N} \cup \{\omega, -1\}$. For any $x \in \mathbb{Z}$ we put $x < \omega$, $\omega + x = x + \omega = \omega$, and for $x \in \mathbb{N}_+$ we put $\omega \cdot x = x \cdot \omega = \omega$. Moreover, $\omega + \omega = \omega - \omega = -\omega + \omega = \omega$, and $\omega \cdot 0 = 0 \cdot \omega = 0$.

For a finite set B, $|B|$ denotes its cardinality, and $\mathcal{P}(B) = \{X \mid X \subseteq B\}$ is the set of all subsets of B. By $\mathbb{N}^B$ we denote the set of all *finite multisets* of elements of B, identifying a *multiset* $M \in \mathbb{N}^B$ with a function $M : B \to \mathbb{N}$; $M(x)$ denotes the *multiplicity* (the number of occurrences) of $x \in B$ in M. As usual, for multisets M_1, M_2 we rather write $M_1 + M_2$ instead of $M_1 \cup M_2$.

By a *partition of a (finite) set* B we understand a set $\mathcal{T} \subseteq \mathcal{P}(B)$ of nonempty pairwise disjoint subsets of B such that their union is B. The elements of a partition $\mathcal{T}$ are its *classes*. The set of all partitions of a set B is denoted by *Partitions*(B). By $\sqsubseteq$ we denote the standard *refinement-order* on *Partitions*(B): $\mathcal{T}_1 \sqsubseteq \mathcal{T}_2$ if each class $T \in \mathcal{T}_1$ is a subset of some class $T' \in \mathcal{T}_2$. We also use the standard notational variants: e.g., $\mathcal{T}_1 \sqsupsetneq \mathcal{T}_2$ denotes that $\mathcal{T}_2$ is a proper refinement of $\mathcal{T}_1$ (hence $\mathcal{T}_2 \sqsubseteq \mathcal{T}_1$ and $\mathcal{T}_2 \neq \mathcal{T}_1$).

Given $\mathcal{T}_1, \mathcal{T}_2 \in$ *Partitions*(B), by $\mathcal{T}_1 \sqcap \mathcal{T}_2$ we denote the *meet* of $\mathcal{T}_1$ and $\mathcal{T}_2$, i.e., the greatest lower bound $\mathcal{T}$ for $\{\mathcal{T}_1, \mathcal{T}_2\}$ w.r.t. $\sqsubseteq$ (hence two elements of B are in the same class of $\mathcal{T}_1 \sqcap \mathcal{T}_2$ iff they are in the same class of $\mathcal{T}_1$ and in the same class of $\mathcal{T}_2$); more generally, $\bigsqcap_{i \in I} \mathcal{T}_i$ is the meet of the set $\{\mathcal{T}_i \mid i \in I\} \subseteq$ *Partitions*(B).

2.2. Labelled transition systems and bisimulation equivalence

A *labelled transition system (LTS)* is a tuple $\mathcal{L} = (S, \mathrm{Act}, (\xrightarrow{a})_{a \in \mathrm{Act}})$ where S is the set of *states*, Act is the set of *actions* and $\xrightarrow{a} \subseteq S \times S$ is the set of *a-transitions* (transitions labelled with a), for each $a \in \mathrm{Act}$. We usually write $s \xrightarrow{a} s'$ instead of $(s, s') \in \xrightarrow{a}$. The relations $\xrightarrow{a}$ are naturally extended to $\xrightarrow{w}$ for $w \in \mathrm{Act}^*$ (i.e., for finite words over Act): we have $s \xrightarrow{\varepsilon} s$ (where ε denotes the empty word), and $s \xrightarrow{a} s'$ and $s' \xrightarrow{w} s''$ entails $s \xrightarrow{aw} s''$. We say that s' is *reachable from* s if $s \xrightarrow{w} s'$ for some $w \in \mathrm{Act}^*$.

[1] The algorithm also yields a smaller degree of the respective polynomial but these technical details are not discussed here; the author thanks to Martin Kot and Zdeněk Sawa for discussions and the works [14,15].

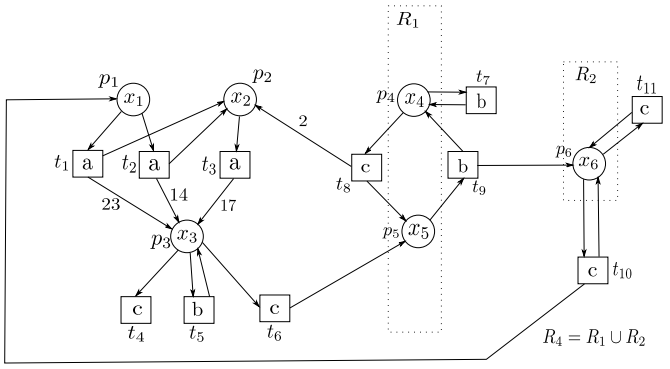

Fig. 1. Example of a BPP-net $\mathrm{N} = (\mathrm{P}, \mathrm{T}, \textsc{pre}, \textsc{post}, \textsc{Act}, \textsc{lab})$, with marking $M = (x_1, x_2, x_3, x_4, x_5, x_6)$.

Given an LTS $\mathcal{L} = (S, \textsc{Act}, (\xrightarrow{a})_{a \in \textsc{Act}})$, a binary relation $\mathcal{R} \subseteq S \times S$ is a *bisimulation* if for each $(s_1, s_2) \in \mathcal{R}$ the following conditions hold:

- if $s_1 \xrightarrow{a} s_1'$, then there is s_2' such that $s_2 \xrightarrow{a} s_2'$ and $(s_1', s_2') \in \mathcal{R}$,
- if $s_2 \xrightarrow{a} s_2'$, then there is s_1' such that $s_1 \xrightarrow{a} s_1'$ and $(s_1', s_2') \in \mathcal{R}$.

Two states s_1, s_2 are *bisimulation equivalent*, or *bisimilar*, which is denoted by $s_1 \sim s_2$, if there is a bisimulation $\mathcal{R}$ such that $(s_1, s_2) \in \mathcal{R}$. It is standard to note that *bisimulation equivalence* (or *bisimilarity*), i.e. the relation $\sim \subseteq S \times S$, is an equivalence relation and the largest bisimulation on S.

Though we have defined bisimulation equivalence on the states of one LTS, it can also relate states of two different LTSs, in which case we implicitly refer to the disjoint union of respective LTSs.

2.3. Basic Parallel Processes (BPP), presented as (markings of) BPP-nets N, and the related LTSs $\mathcal{L}_{\mathrm{N}}$

Basic parallel processes (BPP) can be viewed as a fragment of Milner's calculus of communicating systems (CCS) [4] (without restriction, relabelling and communication). An alternative definition is given in the framework of process rewrite systems [17]. Here we use a technically convenient representation of BPP systems by "BPP-nets", a special type of labelled Petri nets where each transition has exactly one input place.

Formally, a *BPP-net* is a tuple $\mathrm{N} = (\mathrm{P}, \mathrm{T}, \textsc{pre}, \textsc{post}, \textsc{Act}, \textsc{lab})$ where P, T and Act are finite sets of *places*, *transitions*, and *actions* (or *transition-labels*), respectively; $\textsc{pre} : \mathrm{T} \to \mathrm{P}$ is a function attaching a unique *input place* $\textsc{pre}(t)$ to a transition t, $\textsc{post} : \mathrm{T} \to \mathbb{N}^P$ is a function attaching a multiset of *output places* $\textsc{post}(t)$ to a transition t, and $\textsc{lab} : \mathrm{T} \to \textsc{Act}$ labels each transition by an action. The function $\textsc{pre} : \mathrm{T} \to \mathrm{P}$ is also extended to $\textsc{pre} : \mathcal{P}(\mathrm{T}) \to \mathcal{P}(\mathrm{P})$, so that for $T \subseteq \mathrm{T}$ we have $\textsc{pre}(T) = \{\, \textsc{pre}(t) \mid t \in T \,\}$.

An example of a BPP-net $\mathrm{N} = (\mathrm{P}, \mathrm{T}, \textsc{pre}, \textsc{post}, \textsc{Act}, \textsc{lab})$ is shown in Fig. 1, where

$$\mathrm{P} = \{p_1, p_2, p_3, p_4, p_5, p_6\},\ \mathrm{T} = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}\},\ \textsc{Act} = \{\mathrm{a}, \mathrm{b}, \mathrm{c}\},$$

and pre, post, lab are depicted in an obvious way: e.g., for t_2 we have $\textsc{pre}(t_2) = p_1$, $\textsc{post}(t_2) = \{p_2, 14 \cdot p_3\}$, and $\textsc{lab}(t_2) = \mathrm{a}$; the multiplicity of a transition-to-place edge is omitted when it is 1.

A *marking* M of N is a multiset of places, hence a function $M : \mathrm{P} \to \mathbb{N}$. A *transition* t is *enabled* in M if $M(\textsc{pre}(t)) \geq 1$. A transition t that is enabled in M can be *performed*, which results in the marking $M' = M - \{\textsc{pre}(t)\} + \textsc{post}(t)$; we write $M \xrightarrow{t} M'$. E.g., if the marking $M = \{x_1 \cdot p_1, x_2 \cdot p_2, \ldots, x_6 \cdot p_6\}$ in Fig. 1, written as $M = (x_1, x_2, x_3, x_4, x_5, x_6)$ for short, satisfies $x_1 \geq 1$, then we have $M \xrightarrow{t_2} M' = (x_1{-}1, x_2{+}1, x_3{+}14, x_4, x_5, x_6)$.

A BPP net $\mathrm{N} = (\mathrm{P}, \mathrm{T}, \textsc{pre}, \textsc{post}, \textsc{Act}, \textsc{lab})$ generates the LTS $\mathcal{L}_{\mathrm{N}} = (\mathcal{M}(\mathrm{N}), \textsc{Act}, (\xrightarrow{a})_{a \in \textsc{Act}})$ where $\mathcal{M}(\mathrm{N}) = \mathbb{N}^{\mathrm{P}}$ is the set of markings of N, and we have $M \xrightarrow{a} M'$ if $M \xrightarrow{t} M'$ for $t \in \mathrm{T}$ with $\textsc{lab}(t) = a$.

2.4. Problem of deciding bisimilarity on BPP

We are interested in the complexity of the following decision problem:

BPP-Bisim
Instance: a BPP-net N and two markings $M_1, M_2 \in \mathcal{M}(\mathrm{N})$.
Question: is $M_1 \sim M_2$?

In the next section we show that the problem BPP-Bisim is in PSPACE (and thus PSPACE-complete), assuming a standard encoding of problem instances. Since there are numbers in the instances, namely in the presentations of multisets $\text{post}(t)$ (for transitions t of N) and M_1, M_2, we stress that we assume a standard binary (or decimal) presentation of these numbers. We note that Srba [3] showed a PSPACE lower bound for BPP-Bisim that also applies when the numbers are presented in unary.

In the next two subsections we recall some standard notions and facts that are useful in the later proofs.

2.5. Traps, marked traps, $\text{trap}(Q)$, restrictions $M|_Q$

Given a BPP-net $\text{N} = (\text{P}, \text{T}, \text{pre}, \text{post}, \text{Act}, \text{lab})$, we say that $R \subseteq \text{P}$ is a *trap* if for each $t \in \text{T}$ with $\text{pre}(t) \in R$ we have $\text{post}(t) \cap R \neq \emptyset$. E.g., the set $R_1 = \{p_4, p_5\}$ in Fig. 1 is a trap. We note that $\emptyset$ is the least trap and $R = \bigcup \{R' \subseteq \text{P} \mid R' \text{ is a trap }\}$ is the largest trap (since the union of traps is a trap).

Given $Q \subseteq \text{P}$, we can compute the largest trap contained in Q, denoted by $\text{trap}(Q)$, by a standard polynomial-time algorithm:

$R := Q$;
repeat
for each $t \in \text{T}$ such that $\text{pre}(t) \in R$ and $\text{post}(t) \cap R = \emptyset$ **do** $R := R \setminus \{\text{pre}(t)\}$
until R is a trap ($\star$ here $R = \text{trap}(Q)$ $\star$).

By $M|_Q$ we denote the *restriction* of marking M to Q (hence $M|_Q : Q \to \mathbb{N}$ where $M|_Q(p) = M(p)$ for all $p \in Q$). By $\mathbf{0}$ we denote the empty multiset (over a domain clear from context). We say that $Q \subseteq \text{P}$ is *unmarked in* M if $M|_Q = \mathbf{0}$; Q is *marked in* M if $M|_Q \neq \mathbf{0}$ (hence $M(p) \geq 1$ for some $p \in Q$).

We observe that if a trap is marked in M then it is also marked in all M' reachable from M.

2.6. Norms of sets of places, and normed BPP (nBPP)

In our later reasoning, an important role will be played by functions $\text{norm}_Q : \mathcal{M}(\text{N}) \to \mathbb{N}_\omega$, for $Q \subseteq \text{P}$ in a given BPP-net $\text{N} = (\text{P}, \text{T}, \text{pre}, \text{post}, \text{Act}, \text{lab})$. The value $\text{norm}_Q(M)$ is the length of a shortest $w \in \text{Act}^*$ such that $M \xrightarrow{w} M'$ where $M'|_Q = \mathbf{0}$; we put $\text{norm}_Q(M) = \omega$ if there is no such w. In fact, we have already observed that $M|_{\text{trap}(Q)} \neq \mathbf{0}$ entails $\text{norm}_Q(M) = \omega$ (if $\text{trap}(Q)$ is marked in M, then it remains marked in all M' reachable from M).

If we assume that the places in P are ordered, viewing P as a tuple $(p_1, p_2, \ldots, p_k)$, then $\text{norm}_Q(M)$, where M is viewed as a vector $(x_1, x_2, \ldots, x_k) \in \mathbb{N}^k$, can be obviously expressed as a linear function (of type $\mathbb{N}^k \to \mathbb{N}_\omega$) of the form

$$c_1^Q \cdot x_1 + c_2^Q \cdot x_2 \cdots + c_k^Q \cdot x_k$$

for some coefficients $c_i^Q \in \mathbb{N}_\omega$; in fact, $c_i^Q = \text{norm}_Q(\{p_i\})$. E.g., in Fig. 1 for $Q = \{p_1, p_3, p_6\}$ we have $\text{norm}_Q(\{p_2\}) = \text{norm}_Q(\{p_4\}) = \text{norm}_Q(\{p_5\}) = 0$, $\text{norm}_Q(\{p_6\}) = \omega$, $\text{norm}_Q(\{p_3\}) = 1$, and $\text{norm}_Q(\{p_1\}) = 1 + \min\{\text{norm}_Q(\{p_2, 23 \cdot p_3\}), \text{norm}_Q(\{p_2, 14 \cdot p_3\})\} = 1 + \min\{0 + 23 \cdot 1, 0 + 14 \cdot 1\} = 15$. Hence

$$\text{norm}_Q(x_1, x_2, x_3, x_4, x_5, x_6) = 15 \cdot x_1 + 0 \cdot x_2 + 1 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5 + \omega \cdot x_6 = 15 \cdot x_1 + x_3 + \omega \cdot x_6.$$

(Recall that we have put $\omega \cdot 0 = 0$, $\omega \cdot x = \omega$ for all $x \in \mathbb{N}_+$, and $\omega + z = z + \omega = \omega$ for all $z \in \mathbb{Z} \cup \{\omega\}$.)

There is a straightforward polynomial-time algorithm computing the coefficients c_i^Q: if $p_i \in \text{P} \setminus Q$, then $c_i^Q = 0$; if $p_i \in \text{trap}(Q)$, then $c_i^Q = \omega$; computing the coefficients for places $p_i \in Q \setminus \text{trap}(Q)$ is clear from the following inductive fact: $c_i^Q = 1 + \min\{\text{norm}_Q(\text{post}(t)) \mid t \in \text{T},\ \text{pre}(t) = p_i\}$.

A *BPP-net* $\text{N} = (\text{P}, \text{T}, \text{pre}, \text{post}, \text{Act}, \text{lab})$ is *normed* if $\mathbf{0}$ is reachable from each marking $M \in \mathcal{M}(\text{N})$; in other words, if $\text{trap}(\text{P}) = \emptyset$.

3. Bisimilarity on BPP is in PSPACE (and in PTIME on normed BPP)

We start with a sketch of the ideas on which the later polynomial-space algorithm is based; the ideas are illustrated on the BPP-net $\text{N} = (\text{P}, \text{T}, \text{pre}, \text{post}, \text{Act}, \text{lab})$ in Fig. 1.

Let us consider the partition $\text{PartLab} = \{T_a \mid a \in \text{Act}\}$ of the set T, where $T_a = \{t \in \text{T} \mid \text{lab}(t) = a\}$. In our example we have $\text{PartLab} = \{T_\text{a}, T_\text{b}, T_\text{c}\}$ where

$$T_\text{a} = \{t_1, t_2, t_3\},\ T_\text{b} = \{t_5, t_7, t_9\},\ \text{and } T_\text{c} = \{t_4, t_6, t_8, t_{10}, t_{11}\}. \tag{1}$$

The *partition* PartLab is trivially *match-constraining* in the sense that $M_1 \sim M_2$ entails that if $M_1 \xrightarrow{t} M_1'$ then there is t' from the same partition-class as t such that $M_2 \xrightarrow{t'} M_2'$ and $M_1' \sim M_2'$. Hence $M_1 \sim M_2$ also entails that $\text{norm}_{\text{pre}(T_a)}(M_1) = \text{norm}_{\text{pre}(T_a)}(M_2)$ for each $a \in \text{Act}$.

Indeed: If $\text{NORM}_Q(M_1) \neq \text{NORM}_Q(M_2)$ where $Q = \text{PRE}(T_a)$ for some $a \in \text{ACT}$, say $\text{NORM}_Q(M_1) > \text{NORM}_Q(M_2)$, then there is a shortest word $w \in \text{ACT}^*$ such that $M_2 \xrightarrow{w} M_2'$ where $\text{NORM}_Q(M_2') = 0$, and the fact $\text{NORM}_Q(M_1) > \text{NORM}_Q(M_2)$ entails that for all M_1' where $M_1 \xrightarrow{w} M_1'$ we have $\text{NORM}_Q(M_1') \geq 1$; hence each such M_1' enables an a-transition ($M_1'|_{\text{PRE}(T_a)} \neq \mathbf{0}$) while M_2' enables no a-transition ($M_2'|_{\text{PRE}(T_a)} = \mathbf{0}$). Hence $M_2 \xrightarrow{w} M_2'$ cannot be matched by any $M_1 \xrightarrow{w} M_1'$ for which $M_1' \sim M_2'$; this entails that $M_1 \nsim M_2$.

Hence each *set of places* $Q \subseteq \text{P}$ from the collection $C_0 = \{Q \mid Q = \text{PRE}(T_a)$ for some $a \in \text{ACT}\}$ is *important* in the sense that the equality of the values of the function NORM_Q is a necessary condition for two markings to be bisimilar. We thus have

$$M_1 \sim M_2 \Longrightarrow M_1 \equiv_{C_0} M_2$$

where for each $C \subseteq \mathcal{P}(\text{P})$ we put $M_1 \equiv_C M_2 \Leftrightarrow_{df} \text{NORM}_Q(M_1) = \text{NORM}_Q(M_2)$ for all $Q \in C$.

In our example the collection C_0 is $\{Q_1, Q_2, Q_3\}$ where

$$Q_1 = \text{PRE}(T_\text{a}) = \{p_1, p_2\},\ Q_2 = \text{PRE}(T_\text{b}) = \{p_3, p_4, p_5\},\ Q_3 = \text{PRE}(T_\text{c}) = \{p_3, p_4, p_6\}. \tag{2}$$

The relation $\equiv_{C_0}$ is not a bisimulation in general: in our example we have $\{p_4\} \equiv_{C_0} \{p_3, p_5\}$ (since the norms of Q_1, Q_2, Q_3 are, respectively, $0, \omega, 1$ in both markings) but the transition $\{p_3, p_5\} \xrightarrow{c} \{p_5\}$ can only be matched by $\{p_4\} \xrightarrow{c} \{2 \cdot p_2, p_5\}$, and these two transitions cause different changes for NORM_{Q_1} since $\text{NORM}_{Q_1}(\{p_5\}) = 0 \neq \text{NORM}_{Q_1}(\{2 \cdot p_2, p_5\}) = 2$.

Our aim will be to (stepwise) extend the collection C_0 to a collection IS of important sets for which we will achieve $\sim = \equiv_{\text{IS}}$ (hence the equality of the values of NORM_Q for all $Q \in \text{IS}$ will be not only necessary but also sufficient for two markings to be bisimilar). The main idea is captured by the observation that if $M_1 \sim M_2$ and $M_1 \xrightarrow{t} M_1'$, then there is t' such that $\text{LAB}(t') = \text{LAB}(t)$, $M_2 \xrightarrow{t'} M_2'$, and the transitions t, t' cause the same changes of the values of NORM_Q for all important sets $Q \subseteq \text{P}$; this leads to natural refinements of given match-constraining partitions.

We observe that if a set $Q \subseteq \text{P}$ is important in the above sense (hence $\sim \subseteq \equiv_{\{Q\}}$), then also $\text{TRAP}(Q)$, i.e. the largest trap in Q, is important (since $M_1 \sim M_2$ entails that $\text{NORM}_Q(M_1) = \omega$ iff $\text{NORM}_Q(M_2) = \omega$). In our case (2) we have

$$\text{TRAP}(Q_1) = \emptyset,\ \text{TRAP}(Q_2) = R_1 = \{p_4, p_5\},\ \text{TRAP}(Q_3) = R_2 = \{p_6\}.$$

For each important trap R, $M_1 \sim M_2$ simply entails that R is either marked in both M_1 and M_2 or unmarked in both of them. We note that the union of two important traps is an important trap; hence $R_4 = R_1 \cup R_2$ is also important. In our example there is one more trap, namely $R_3 = \{p_5, p_6\}$; it will later turn out to be important as well. Generally, there might be traps that are not important: a simple example arises by adding a place p_7 and a transition t_{12} to Fig. 1, with $\text{PRE}(t_{12}) = p_7$, $\text{POST}(t_{12}) = \{p_7\}$, $\text{LAB}(t_{12}) = \text{c}$; the markings $\{p_6, p_7\}$ and $\{p_6\}$ are then clearly bisimilar, but the trap $\{p_7\}$ is marked in one of them and unmarked in the other.

The idea of the presented algorithm deciding BPP-BISIM is to stepwise refine the match-constraining partition PARTLAB: if two transitions in a current partition-class cause different changes on the norm of some important set, these transitions get separated in the refined partition. The refined partition might then reveal further important sets, which might lead to a further refinement of the current match-constraining partition, etc.; this process must obviously reach a fixpoint, which presents the required collection IS for which $\sim = \equiv_{\text{IS}}$. But there is a subtle point: it is intuitively clear that the mentioned fixpoint-partition will depend on which important traps are currently marked (i.e., for which important sets Q we have that the value of NORM_Q is ω); in our example in Fig. 1, the a-transitions t_1 and t_2 cause different changes on NORM_{Q_2} ($Q_2 = \text{PRE}(T_\text{b})$), namely 23 and 14. It thus seems that t_1, t_2 should be separated in the final partition; but this is not true when NORM_{Q_2} is ω (i.e., when $\text{TRAP}(Q_2)$ is marked), in which case t_1 and t_2 do not influence the value of NORM_{Q_2} since this remains ω forever.

Hence we will have more "fixpoint-partitions": each important trap R will have a related partition TRAPPART_R. The partition TRAPPART_R will be the intended match-constraining partition in the case when R is the *largest unmarked important trap*. The fact that there can be exponentially many important traps (e.g., the union of traps chosen out of possibly many small important traps is an important trap) makes the algorithm exponential, though realizable in polynomial space as we will show.

Let us now try to explore for which M_1, M_2 we have $M_1 \sim M_2$ in the example in Fig. 1 in the simplest case, i.e. when the largest unmarked important trap is $\emptyset$ (in other words, when all nonempty important traps are marked). So we aim to find when $M_1 \sim M_2$ where

$$M_1 = (x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}) \text{ and } M_2 = (x_{21}, x_{22}, x_{23}, x_{24}, x_{25}, x_{26}) \tag{3}$$

(i.e., $M_1(p_i) = x_{1i}$ and $M_2(p_i) = x_{2i}$), assuming that both R_1 and R_2 are marked in M_1 as well as in M_2 (i.e., $M_1|_{R_1} \neq \mathbf{0}$, $M_1|_{R_2} \neq \mathbf{0}$, $M_2|_{R_1} \neq \mathbf{0}$, $M_2|_{R_2} \neq \mathbf{0}$). In this case the norms of $Q_2 = \text{PRE}(T_\text{b})$ and $Q_3 = \text{PRE}(T_\text{c})$ are ω in all markings reachable from M_1 and M_2, so we can concentrate just on keeping the equality $\text{NORM}_{Q_1}(M_1) = \text{NORM}_{Q_1}(M_2)$ for $Q_1 =$

$\text{PRE}(T_a) = \{p_1, p_2\}$. Let us look at the changes $\delta_{Q_1}(t)$ caused by transitions t on the value of NORM_{Q_1}, hence $\delta_{Q_1}(t) = \text{NORM}_{Q_1}(\text{POST}(t_1)) - \text{NORM}_{Q_1}(\{\text{PRE}(t_1)\})$; these changes are given in the next two lines.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}
δ_{Q_1}	-1	-1	-1	0	0	0	0	2	0	2	0

The values $\delta_{Q_1}(t)$ give us the partition

$$\text{PARTCHANGE}_{Q_1} = \{\{t_1, t_2, t_3\}, \{t_4, t_5, t_6, t_7, t_9, t_{11}\}, \{t_8, t_{10}\}\}.$$

Hence we refine the initial match-constraining partition PARTLAB (1) by PARTCHANGE_{Q_1}, which yields $\mathcal{T} = \text{PARTLAB} \sqcap \text{PARTCHANGE}_{Q_1} = \{T_1, T_2, T_3, T_4\}$ where

$$T_1 = \{t_1, t_2, t_3\},\ T_2 = \{t_5, t_7, t_9\},\ T_3 = \{t_4, t_6, t_{11}\},\ T_4 = \{t_8, t_{10}\}. \tag{4}$$

Besides Q_1, Q_2, Q_3 from (2), we have thus got the new important sets

$$Q_4 = \text{PRE}(T_3) = \{p_3, p_6\} \text{ and } Q_5 = \text{PRE}(T_4) = \{p_4, p_6\};$$

this was related to the assumption that the trap $\emptyset$ is unmarked, which holds trivially. Since $\text{TRAP}(Q_4) = \text{TRAP}(Q_5) = \{p_6\} = R_2$, we have got no new important traps. Moreover, in our current case, which assumes that $\emptyset$ is the largest unmarked important trap, we also have that the values of NORM_{Q_4} and NORM_{Q_5} are ω; hence the partition $\mathcal{T} = \{T_1, T_2, T_3, T_4\}$ (4) is a fixpoint partition (since among NORM_{Q_i}, $i \in \{1, 2, 3, 4, 5\}$, only NORM_{Q_1} is finite, and $\mathcal{T} \sqcap \text{PARTCHANGE}_{Q_1} = \mathcal{T}$), and it is thus the partition $\text{TRAPPART}_\emptyset$.

In fact, we have shown that if $\text{NORM}_{\text{PRE}(T_i)}(M_1) = \text{NORM}_{\text{PRE}(T_i)}(M_2) = \omega$ for all $i \in \{2, 3, 4\}$, then the equality $\text{NORM}_{\text{PRE}(T_1)}(M_1) = \text{NORM}_{\text{PRE}(T_1)}(M_2)$ is not only necessary but also sufficient for M_1, M_2 to be bisimilar. We can verify that the relation

$$\mathcal{R} = \{\, (M_1, M_2) \mid R_1 \text{ and } R_2 \text{ are marked in } M_1 \text{ and in } M_2,\ \text{NORM}_{Q_1}(M_1) = \text{NORM}_{Q_1}(M_2) \,\}$$

is a bisimulation; we note that $\text{NORM}_{Q_1}(M_1) = \text{NORM}_{Q_1}(M_2)$ means $2x_{11} + x_{12} = 2x_{21} + x_{22}$ in the notation (3). In particular, if $(M_1, M_2) \in \mathcal{R}$ and $M_1 \xrightarrow{t} M_1'$, then there is t' from the same partition-class of $\mathcal{T}$ (4) as t such that $M_2 \xrightarrow{t'} M_2'$ where $(M_1', M_2') \in \mathcal{R}$.

Now we look at the case where the largest unmarked important trap is $R_2 = \{p_6\}$; in this situation also the important traps that are proper subsets of R_2 are unmarked. To construct TRAPPART_{R_2}, we thus start with the meet ($\sqcap$) of the partitions TRAPPART_R where R are the important traps that are proper subsets of R_2; in our case we have just one such trap R, namely $R_0 = \emptyset$. We thus start with the partition $\mathcal{T} = \{T_1, T_2, T_3, T_4\}$ given by (4), which we aim to refine by PARTCHANGE_Q for those sets $Q = R_2 \cup \text{PRE}(T_i)$, $i \in \{1, 2, 3, 4\}$, for which NORM_Q is finite in the current situation (when R_2 is the largest unmarked important trap), hence for which $\text{TRAP}(R_2 \cup \text{PRE}(T_i)) = R_2$; this holds for $R_2 \cup \text{PRE}(T_1) = \{p_1, p_2, p_6\}$, $R_2 \cup \text{PRE}(T_3) = \{p_3, p_6\}$, and $R_2 \cup \text{PRE}(T_4) = \{p_4, p_6\}$ (but not for $R_2 \cup \text{PRE}(T_2) = \{p_3, p_4, p_5, p_6\}$).

E.g., the class $T_1 = \{t_1, t_2, t_3\}$ breaks down to the singleton classes $\{t_1\}$, $\{t_2\}$, $\{t_3\}$ since $\delta_Q(t_1) = 23$, $\delta_Q(t_2) = 14$, $\delta_Q(t_3) = 17$ for $Q = R_2 \cup \text{PRE}(T_3) = \{p_3, p_6\}$. We stress that this does not entail that, e.g., $\text{PRE}(\{t_1\}) = \{p_1\}$ is an important set; since we have got the partition-class $T = \{t_1\}$ under the assumption that R_2 is unmarked, we only deduce that $R_2 \cup \text{PRE}(T) = \{p_1, p_6\}$ is an important set (and $Q = R_2 \cup \text{PRE}(T)$ can serve for further partition-refinements by δ_Q, to construct TRAPPART_{R_2}, since $\text{TRAP}(R_2 \cup \text{PRE}(T)) = R_2$).

We can also note that a refinement of the class $T_2 = \{t_5, t_7, t_9\}$ in (4) yields the singleton class $\{t_9\}$ since $\delta_Q(t_9) = \omega$ while $\delta_Q(t_5) = \delta_Q(t_7) = 0$ for $Q = R_2 \cup \text{PRE}(T_1) = \{p_1, p_2, p_6\}$ (as well as for other relevant Q). The new important set $R_2 \cup \text{PRE}(\{t_9\}) = \{p_5, p_6\}$ also yields a new important trap, namely $R_3 = \text{TRAP}(\{p_5, p_6\}) = \{p_5, p_6\}$, which is a proper superset of $R_2 = \{p_6\}$; this new important trap is marked in the currently considered situation (when R_2 is the largest unmarked important trap), hence δ_Q for $Q = R_2 \cup \text{PRE}(\{t_9\})$ will play no role in constructing TRAPPART_{R_2}.

Now we make the above intuitively described process formal, for a fixed general BPP-net

$$\text{N} = (\text{P}, \text{T}, \text{PRE}, \text{POST}, \text{ACT}, \text{LAB}).$$

Definition 1. (Partitions PARTLAB, PARTCHANGE_Q, $\text{REFINE}_R(\mathcal{T})$, $\text{STABILIZE}_R(\mathcal{T})$ of the set T of transitions)

1. $\text{PARTLAB} = \{\, T_a \mid a \in \text{ACT} \,\}$ where $T_a = \{\, t \in \text{T} \mid \text{LAB}(t) = a \,\}$.
2. For $Q \subseteq \text{P}$ we define:
   - $\delta_Q : \text{T} \to \mathbb{N}_{\omega,-1}$ where $\delta_Q(t) = \text{NORM}_Q(\text{POST}(t)) - \text{NORM}_Q(\text{PRE}(t))$ (using our conventions for ω that include $\omega - \omega = \omega$);
   - PARTCHANGE_Q is the partition of T such that t_1 and t_2 are in the same class iff $\delta_Q(t_1) = \delta_Q(t_2)$.
3. Given a trap $R \subseteq \text{P}$ and a partition $\mathcal{T}$ of T, we define:

- $\textsc{Refine}_R(\mathcal{T}) = \mathcal{T} \sqcap \prod \{\textsc{PartChange}_Q \mid Q = R \cup \textsc{pre}(T),\ T$ is a class of $\mathcal{T}$, and $\textsc{trap}(Q) = R\}$ (hence we do not use $R \cup \textsc{pre}(T)$ where $\textsc{trap}(R \cup \textsc{pre}(T))$ is larger than R);
- $\textsc{Stabilize}_R(\mathcal{T})$ is the partition $\mathcal{T}'$ that is *stable*, i.e. $\mathcal{T}' = \textsc{Refine}_R(\mathcal{T}')$, and occurs in the sequence $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \ldots$ where $\mathcal{T}_0 = \mathcal{T}$ and $\mathcal{T}_i = \textsc{Refine}_R(\mathcal{T}_{i-1})$ for $i = 1, 2, \ldots$.

In particular, in the normed case, when N is an nBPP-net, we have $\textsc{trap}(\mathrm{P}) = \emptyset$. In this case, our later more general result (namely Theorem 3) entails that $M_1 \sim M_2$ iff $\textsc{norm}_Q(M_1) = \textsc{norm}_Q(M_2)$ for all $Q = \textsc{pre}(T)$ where T is a class of the partition $\textsc{Stabilize}_\emptyset(\textsc{PartLab})$. In this case, the respective algorithm deciding $M_1 \sim M_2$ obviously works in *polynomial time*, since it just constructs the sequence $\mathcal{T}_0 \supsetneq \mathcal{T}_1 \supsetneq \mathcal{T}_2, \ldots, \supsetneq \mathcal{T}_m$ where $\mathcal{T}_0 = \textsc{PartLab}$, $\mathcal{T}_i = \textsc{Refine}_\emptyset(\mathcal{T}_{i-1})$ for $i = 1, 2, \ldots, m$, and $\mathcal{T}_m = \textsc{Refine}_\emptyset(\mathcal{T}_m) = \textsc{Stabilize}_\emptyset(\textsc{PartLab})$, and checks if $\textsc{norm}_Q(M_1) = \textsc{norm}_Q(M_2)$ for all $Q = \textsc{pre}(T)$ where T is a class of $\mathcal{T}_m$.

We recall that $M \xrightarrow{t} M'$ entails $\textsc{norm}_Q(M') = \textsc{norm}_Q(M) + \delta_Q(t)$. The value $\delta_Q(t) \in \mathbb{N}_{\omega,-1}$ is thus relevant only when $\textsc{norm}_Q(M)$ is finite; we can view the partition $\textsc{PartChange}_Q$ as irrelevant for M (and all markings reachable from M) when $\textsc{norm}_Q(M) = \omega$. This is reflected by the condition $\textsc{trap}(Q) = R$ in defining $\textsc{Refine}_R(\mathcal{T})$ in the point 3 of Definition 1 (here $\textsc{trap}(Q) = R$ is equivalent to $\textsc{trap}(Q) \subseteq R$ since $Q \supseteq R$), which should be viewed as capturing the markings where R is the largest important trap that is unmarked.

Definition 2 below formalizes the notions of "important traps" $R \subseteq \mathrm{P}$, and "match-constraining partitions" $\textsc{TrapPart}_R$ of T; the definition uses mutually related inductions, and also yields the notion of "important sets of places" (of the form $Q = R \cup \textsc{pre}(T)$ where T is a class of $\textsc{TrapPart}_R$). The aim of the introduced notions is clear from Theorem 3, on which a polynomial-space algorithm is built, as is shown in the proof of Theorem 4. The definition should be understandable due to the previous informal discussion; we stress that we indeed have $\sim \subseteq \equiv_{\{Q\}}$ for all important sets Q but we do not claim that $\sim \subseteq \equiv_{\{Q\}}$ entails that Q is important in the sense of the following technical definition.

Definition 2. (Important traps R, partitions $\textsc{TrapPart}_R$, important sets $Q = R \cup \textsc{pre}(T)$)

- *Important traps* $R \subseteq \mathrm{P}$ and *important sets* $Q \subseteq \mathrm{P}$ are defined inductively as follows:
  - $\emptyset$ is an important trap;
  - if R is an important trap and T is a class of $\textsc{TrapPart}_R$, then $Q = R \cup \textsc{pre}(T)$ is an important set, also called an *important set related to* R, and $\textsc{trap}(Q)$ is an important trap;
  - if R_1 and R_2 are important traps, then $R_1 \cup R_2$ is an important trap.
- If $R \subseteq \mathrm{P}$ is an important trap, then the *match-constraining partition related to* R, denoted by $\textsc{TrapPart}_R$, is the following partition of the set T of transitions:

$$\textsc{TrapPart}_R = \textsc{Stabilize}_R(\textsc{InitPart}_R) \text{ where}$$
$$\textsc{InitPart}_R = \textsc{PartLab} \sqcap \prod_{Q \in \mathrm{IS}_{\subsetneq R}} \textsc{PartChange}_Q \text{ and}$$
$$\mathrm{IS}_{\subsetneq R} = \{\, Q \mid Q \text{ is an important set related to some } R' \subsetneq R, \text{ and } \textsc{trap}(Q) \subseteq R \,\}.$$

The chosen form of the defined notions should serve for smooth proofs of Theorems 3 and 4.

We recall that for each set $Q \subseteq \mathrm{P}$ we have $\textsc{norm}_Q(M) < \omega$ iff $M|_{\textsc{trap}(Q)} = \mathbf{0}$. Hence the set

$$\mathrm{F}_M = \{\, Q \subseteq \mathrm{P} \mid Q \text{ is an important set, } \textsc{norm}_Q(M) < \omega\} \tag{5}$$

of the important sets with finite norms in M is tightly related to the largest important trap that is unmarked in M, which is the set

$$R_M = \bigcup_{Q \in \mathrm{F}_M} \textsc{trap}(Q). \tag{6}$$

We note that F_M and R_M can only get smaller by transition performing: if $M \xrightarrow{t} M'$, then $\mathrm{F}_M \supseteq \mathrm{F}_{M'}$ and $R_M \supseteq R_{M'}$ (which is related to the fact that if $M \xrightarrow{t} M'$ and $\textsc{norm}_Q(M) = \omega$, then $\textsc{norm}_Q(M') = \omega$). We will prove that $M_1 \sim M_2$ entails that $\mathrm{F}_{M_1} = \mathrm{F}_{M_2}$, and thus $R_{M_1} = R_{M_2}$. The aim of the (match-constraining) partition $\textsc{TrapPart}_R$ is to capture the matching pairs of moves $M_1 \xrightarrow{t_1} M_1'$ and $M_2 \xrightarrow{t_2} M_2'$, where $\textsc{lab}(t_1) = \textsc{lab}(t_2)$ and $M_1' \sim M_2'$, for the pairs $M_1 \sim M_2$ where $R_{M_1} = R_{M_2} = R$. It is thus clear that for $R' \subseteq R$ we should have $\textsc{TrapPart}_{R'} \sqsupseteq \textsc{TrapPart}_R$, since the match-constraints are stronger when the set of important sets with finite norms is larger. In more detail, the inequality $\delta_Q(t_1) \neq \delta_Q(t_2)$ matters in the situations when the norm of Q is finite but it does not matter when this norm is infinite. This is the reason why $\textsc{PartChange}_Q$ is used carefully: both $\mathrm{IS}_{\subsetneq R}$ in Definition 2 and $\textsc{Refine}_R(\mathcal{T})$ in Definition 1 require that $\textsc{trap}(Q) \subseteq R$ (and thus the norm of Q is finite in the intended situations where R is unmarked).

The above intuition is now formalized in the proofs of the announced theorems.

Theorem 3. *Given a BPP-net* $\mathrm{N} = (\mathrm{P}, \mathrm{T}, \textsc{pre}, \textsc{post}, \textsc{Act}, \textsc{lab})$*, for all markings* M_1, M_2 *we have*

$$M_1 \sim M_2 \text{ iff } \textsc{norm}_Q(M_1) = \textsc{norm}_Q(M_2) \text{ for all important sets } Q \subseteq \mathrm{P}.$$

Proof. We put $\mathrm{IS} = \{\, Q \subseteq \mathrm{P} \mid Q \text{ is an important set} \,\}$ and define the relation $\equiv_{\mathrm{IS}}$ on the set of markings:

$$M_1 \equiv_{\mathrm{IS}} M_2 \Leftrightarrow_{df} \textsc{norm}_Q(M_1) = \textsc{norm}_Q(M_2) \text{ for all } Q \in \mathrm{IS}.$$

We thus need to show that $\sim \,= \equiv_{\mathrm{IS}}$, i.e.,

$$M_1 \sim M_2 \text{ iff } M_1 \equiv_{\mathrm{IS}} M_2.$$

We show the "if" part (soundness) and the "only-if" part (completeness) separately.

"If" (soundness, i.e., $\equiv_{\mathrm{IS}} \subseteq \sim$*).*

We will show that $\equiv_{\mathrm{IS}}$ is a bisimulation, which entails that $\equiv_{\mathrm{IS}} \subseteq \sim$. Let

$$M_1 \equiv_{\mathrm{IS}} M_2, \text{ and } M_1 \xrightarrow{t_1} M_1';$$

we will be done once we show that there is a transition t_2 such that $\textsc{lab}(t_1) = \textsc{lab}(t_2)$ and $M_2 \xrightarrow{t_2} M_2'$ where $M_1' \equiv_{\mathrm{IS}} M_2'$. Since $M_1 \equiv_{\mathrm{IS}} M_2$, we have $\mathrm{F}_{M_1} = \mathrm{F}_{M_2}$ and $R_{M_1} = R_{M_2}$ (recall (5) and (6)). We put $\mathrm{F} = \mathrm{F}_{M_1} = \mathrm{F}_{M_2}$, $R = R_{M_1} = R_{M_2}$, and note that

$$R = \bigcup_{Q \in \mathrm{F}} \textsc{trap}(Q) \text{ and } M_1|_R = M_2|_R = \mathbf{0}.$$

(R is the largest unmarked important trap both in M_1 and in M_2.)

Let T be the class of $\textsc{TrapPart}_R$ containing t_1 (for which $M_1 \xrightarrow{t_1} M_1'$). For the important set

$$Q_1 = R \cup \textsc{pre}(T) \quad (\text{where } t_1 \in T)$$

we thus have $\textsc{norm}_{Q_1}(M_1) = \textsc{norm}_{Q_1}(M_2) \geq 1$, since $M_1(p) \geq 1$ for some $p \in \textsc{pre}(T)$ due to the fact that t_1 is enabled in M_1. Since $\textsc{norm}_{Q_1}(M_2) \geq 1$ and $M_2|_R = \mathbf{0}$, there is $p' \in \textsc{pre}(T)$ such that $M_2(p') \geq 1$; we can thus fix some $t_2 \in T$ that is enabled in M_2. Since $\textsc{TrapPart}_R \sqsubseteq \textsc{PartLab}$, we have $\textsc{lab}(t_1) = \textsc{lab}(t_2)$.

Let M_2' be the marking for which $M_2 \xrightarrow{t_2} M_2'$; we will show $M_1' \equiv_{\mathrm{IS}} M_2'$, by which the proof will be finished. To this aim we consider an arbitrary set $Q' \in \mathrm{IS}$, hence

$$Q' = R' \cup \textsc{pre}(T')$$

where R' is an important trap and T' is a class in $\textsc{TrapPart}_{R'}$, and show that $\textsc{norm}_{Q'}(M_1') = \textsc{norm}_{Q'}(M_2')$:

- If $Q' \in \mathrm{IS} \setminus \mathrm{F}$ (hence $\textsc{trap}(Q')$ is marked in M_1 and in M_2), then the claim is trivial since $\textsc{norm}_{Q'}(M_1) = \textsc{norm}_{Q'}(M_2) = \textsc{norm}_{Q'}(M_1') = \textsc{norm}_{Q'}(M_2') = \omega$.
- If $Q' \in \mathrm{F}$ (hence $\textsc{norm}_{Q'}(M_1) = \textsc{norm}_{Q'}(M_2) < \omega$, and thus $M_1|_{\textsc{trap}(Q')} = M_2|_{\textsc{trap}(Q')} = \mathbf{0}$), then $R' \subseteq \textsc{trap}(Q') \subseteq R$, and this entails that $\textsc{TrapPart}_R \sqsubseteq \textsc{PartChange}_{Q'}$. Indeed:
  - either $R' \subsetneq R$, in which case $Q' \in \mathrm{IS}_{\subsetneq R}$ and thus $\textsc{InitPart}_R \sqsubseteq \textsc{PartChange}_{Q'}$ [by Definition 2], and the fact that $\textsc{TrapPart}_R \sqsubseteq \textsc{InitPart}_R$ [by Definitions 1 and 2] entails that $\textsc{TrapPart}_R \sqsubseteq \textsc{PartChange}_{Q'}$,
  - or $R' = \textsc{trap}(Q') = R$, hence $T' \in \textsc{TrapPart}_R$, and we have $\textsc{TrapPart}_R \sqsubseteq \textsc{PartChange}_{Q'}$ since $\textsc{TrapPart}_R$ is stable ($\textsc{TrapPart}_R = \textsc{Refine}_R(\textsc{TrapPart}_R)$).

  Hence $\delta_{Q'}(t_1) = \delta_{Q'}(t_2)$, which entails that

$$\textsc{norm}_{Q'}(M_1') = \textsc{norm}_{Q'}(M_1) + \delta_{Q'}(t_1) = \textsc{norm}_{Q'}(M_2) + \delta_{Q'}(t_2) = \textsc{norm}_{Q'}(M_2').$$

"Only-if" (completeness, i.e., $\sim \subseteq \equiv_{\mathrm{IS}}$*).*

We show the implication $M_1 \sim M_2 \Rightarrow M_1 \equiv_{\mathrm{IS}} M_2$ by using a structural induction based on Definitions 2 and 1. First we recall that for each important trap R the definitions yield the sequence

$$\textsc{InitPart}_R = \mathcal{T}_{R,0} \supsetneq \mathcal{T}_{R,1} \supsetneq \mathcal{T}_{R,2} \cdots \supsetneq \mathcal{T}_{R,m_R} = \textsc{TrapPart}_R$$

where $m_R \geq 0$ and $\mathcal{T}_{R,i} = \textsc{Refine}_R(\mathcal{T}_{R,i-1})$ for $i = 1, 2, \ldots, m$ (and $\mathcal{T}_{R,m_R} = \textsc{Refine}_R(\mathcal{T}_{R,m_R})$). This leads us to an extension of the notion of important sets: we say that $Q \subseteq \mathrm{P}$ is an *ext-important set* if it *has an* (R, j)*-presentation*, i.e., $Q = R \cup \textsc{pre}(T)$ where R is an important trap and $T \in \mathcal{T}_{R,j}$ (hence $0 \leq j \leq m_R$).

There is a natural order on the set of presentations of ext-important sets: we say that

$$(R', j') \text{ is smaller than } (R, j) \text{ if either } R' \subsetneq R, \text{ or } R' = R \text{ and } j' < j.$$

We will now show that

$$M_1 \sim M_2 \text{ entails } \textsc{norm}_Q(M_1) = \textsc{norm}_Q(M_2) \text{ for all ext-important sets } Q\,. \tag{7}$$

Our desired implication $M_1 \sim M_2 \Rightarrow M_1 \equiv_{\mathrm{is}} M_2$ is then a corollary.

We prove (7) by contradiction. Let

$$(R, j) \text{ be a minimal presentation for which there are } M_1, M_2, \text{ and } Q = R \cup \textsc{pre}(T),\ T \in \mathcal{T}_{R,j},$$

such that

$$M_1 \sim M_2 \text{ and } \textsc{norm}_Q(M_1) \neq \textsc{norm}_Q(M_2);$$

w.l.o.g. we assume $\textsc{norm}_Q(M_1) > \textsc{norm}_Q(M_2)$. Since $\textsc{norm}_Q(M_2)$ is finite, there is a shortest word $w \in \textsc{Act}^*$ such that $M_2 \xrightarrow{w} M_2'$ where $\textsc{norm}_Q(M_2') = 0$. By the assumption $M_1 \sim M_2$ we derive that there is M_1' such that $M_1 \xrightarrow{w} M_1'$ and $M_1' \sim M_2'$; since $\textsc{norm}_Q(M_1) > \textsc{norm}_Q(M_2)$, our choice of (shortest) w entails that $\textsc{norm}_Q(M_1') \geq 1$. Hence we can safely choose M_1 and M_2 so that

$$M_1 \sim M_2 \text{ and } \textsc{norm}_Q(M_1) > \textsc{norm}_Q(M_2) = 0, \text{ for the above set } Q = R \cup \textsc{pre}(T).$$

We now show that $M_1|_R = M_2|_R = \mathbf{0}$, and M_1 enables some $t \in T$, while M_2 enables no $t \in T$:

- Since $\textsc{norm}_Q(M_2) = 0$, we have $M_2|_{R \cup \textsc{pre}(T)} = \mathbf{0}$; in particular, no $t \in T$ is enabled in M_2.
- Since $\textsc{norm}_Q(M_1) > 0$, we have $M_1|_{R \cup \textsc{pre}(T)} \neq \mathbf{0}$.
  If there was some $p \in R$ such that $M_1(p) \geq 1$, then we had $p \in \textsc{trap}(Q') \subseteq R$ for some important set $Q' \in \mathrm{IS}_{\subsetneq R}$, where $Q' = R' \cup \textsc{pre}(T')$ for an important trap $R' \subsetneq R$ and $T' \in \textsc{TrapPart}_{R'}$. (This follows easily from the inductive definition of important traps [Definition 2].) But this would entail that $\textsc{norm}_{Q'}(M_1) = \omega$ and $\textsc{norm}_{Q'}(M_2) < \omega$, hence $\textsc{norm}_{Q'}(M_1) \neq \textsc{norm}_{Q'}(M_2)$, and the set Q' with its $(R', m_{R'})$-presentation would thus violate our minimality assumption on (R, j) (since $(R', m_{R'})$ related to Q' is smaller than (R, j) related to Q).
  Hence $M_1|_R = \mathbf{0}$ and $M_1|_{\textsc{pre}(T)} \neq \mathbf{0}$, which entails that some $t \in T$ is enabled in M_1.

We fix some $t_1 \in T$ that is enabled in M_1; let $M_1 \xrightarrow{t_1} M_1'$. Since $M_1 \sim M_2$, there must be a matching move $M_2 \xrightarrow{t_2} M_2'$ where $\textsc{lab}(t_1) = \textsc{lab}(t_2)$ and $M_1' \sim M_2'$. Since $t_2 \notin T$, we get that

$$t_1, t_2 \text{ belong to the same class of } \mathcal{T}_{\emptyset,0} = \textsc{PartLab} \text{ but to different classes of } \mathcal{T}_{R,j}. \tag{8}$$

We now show a desired contradiction, namely the existence of an ext-important set Q' such that

1. $Q' = R' \cup \textsc{pre}(T')$ where R' is an important trap and $T' \in \mathcal{T}_{R',j'}$;
2. (R', j') is smaller than (R, j);
3. $\textsc{norm}_{Q'}(M_1) = \textsc{norm}_{Q'}(M_2) < \omega$ (for the above chosen $M_1 \sim M_2$);
4. $\delta_{Q'}(t_1) \neq \delta_{Q'}(t_2)$ (i.e., t_1, t_2 belong to different classes of $\textsc{PartChange}_{Q'}$).

This indeed yields a contradiction, since $M_1' \sim M_2'$ and the value $\textsc{norm}_{Q'}(M_1') = \textsc{norm}_{Q'}(M_1) + \delta_{Q'}(t_1)$ differs from the value $\textsc{norm}_{Q'}(M_2') = \textsc{norm}_{Q'}(M_2) + \delta_{Q'}(t_2)$, which contradicts with the minimality assumption on (R, j).

The existence of Q' satisfying conditions $1-4$ follows by a simple inspection of Definitions 1 and 2 and the following analysis, where we consider two separate cases, a) and b), while recalling the fact (8).

a) Transitions t_1, t_2 belong to different classes of $\textsc{InitPart}_R$ (i.e., of $\mathcal{T}_{R,0}$).
   In this case $R \neq \emptyset$ (recall (8)), and t_1, t_2 belong to different classes of $\textsc{PartChange}_{Q'}$ for some $Q' = R' \cup \textsc{pre}(T')$ where $R' \subsetneq R$, R' is an important trap, $T' \in \textsc{TrapPart}_{R'}$, and $\textsc{trap}(Q') \subseteq R$. (Recall $\textsc{InitPart}_R$ in Definition 2.)
   Since $(R', m_{R'})$ is smaller than (R, j), $\textsc{trap}(Q') \subseteq R$, and $M_1|_R = M_2|_R = \mathbf{0}$, we have $\textsc{norm}_{Q'}(M_1) = \textsc{norm}_{Q'}(M_2) < \omega$ (where the equality follows from our minimality assumption on (R, j)).
b) There is the largest j' such that t_1, t_2 belong to the same class of $\mathcal{T}_{R,j'}$.
   By (8) we have $j' < j$, and t_1, t_2 belong to different classes of $\textsc{Refine}_R(\mathcal{T}_{R,j'})$; this entails that $\delta_{Q'}(t_1) \neq \delta_{Q'}(t_2)$ for some $Q' = R \cup \textsc{pre}(T')$ where $T' \in \mathcal{T}_{R,j'}$ and $\textsc{trap}(Q') = R$. (Recall $\textsc{Refine}_R(\mathcal{T})$ in Definition 1.) (R, j') is smaller than (R, j), and we deduce $\textsc{norm}_{Q'}(M_1) = \textsc{norm}_{Q'}(M_2) < \omega$ as in a). □

Theorem 4. *The bisimilarity problem on BPP (the problem* BPP-Bisim*) is in* PSPACE.

Proof. Theorem 3 suggests a straightforward algorithm to decide whether $M_1 \sim M_2$, when given a BPP-net $\mathrm{N} = (\mathrm{P}, \mathrm{T}, \textsc{pre}, \textsc{post}, \textsc{Act}, \textsc{lab})$ and two markings M_1, M_2:

```
ImpTrap(R): (⋆ returns true if R ⊆ P is an important trap, and false otherwise ⋆)
    if R = ∅ then return true
    for all R1, R2 ⊊ R do
        if R1 ∪ R2 = R and ImpTrap(R1) and ImpTrap(R2) then return true
    for each R' ⊊ R such that ImpTrap(R') do
        T' := TrapPart(R')
        for each T' ∈ T' do
            if R = trap(R' ∪ pre(T')) then return true
    return false

TrapPart(R): (⋆ given an important trap R, it returns TrapPart_R ⋆)
    T := PartLab
    for each R' ⊊ R such that ImpTrap(R') do
        T' := TrapPart(R')
        for each T' ∈ T' do
            Q := R' ∪ pre(T')
            if trap(Q) ⊆ R then T := T ⊓ PartChange_Q
    (⋆ now the program variable T contains the value InitPart_R ⋆)
    return Stabilize_R(T)
```

Fig. 2. Pseudocodes of two crucial procedures.

check if $\mathrm{NORM}_Q(M_1) = \mathrm{NORM}_Q(M_2)$ for all important sets Q.

A question is if such an algorithm can be implemented to run in polynomial space.

In fact, Definitions 2 and 1 describe an inductive construction of all important sets, by constructing all important traps R and their related partitions $\mathrm{TRAPPART}_R$, from the smaller traps with coarser partitions to the larger traps with finer partitions. A direct implementation, constructing and remembering all important traps R and their related partitions $\mathrm{TRAPPART}_R$, yields in general an exponential-time algorithm that also uses exponential space (in the worst case); indeed: there might be many small traps for which we have to consider the union of each collection of these traps.

As already explicitly or implicitly discussed, the operations like computing $\mathrm{NORM}_Q(M)$ and partitions PARTLAB, $\mathrm{PARTCHANGE}_Q$, $\mathrm{REFINE}_R(\mathcal{T})$, as well as $\mathrm{STABILIZE}_R(\mathcal{T})$ are performable in polynomial time. Hence the crucial problem is to generate (and check) all important sets $Q = R \cup \mathrm{PRE}(T)$, by constructing all important traps R and their related partitions $\mathrm{TRAPPART}_R$, while not storing all of them in memory.

We use a standard trick: instead of remembering all computed important traps and their related partitions, these are (re)computed repeatedly in the situations when they are really needed. This can be realized by two interdependent recursive procedures:

- one boolean procedure $\mathrm{IMPTRAP}(R)$ returning *true* for $R \subseteq \mathrm{P}$ iff R is an important trap, and
- a procedure $\mathrm{TRAPPART}(R)$ returning the partition $\mathrm{TRAPPART}_R$ of the set T of transitions, when given an important trap R.

The procedures are described by the pseudocodes in Fig. 2.

We note that the depth of recursion in the invocations of IMPTRAP and TRAPPART is at most |P|: the argument $R' \subseteq \mathrm{P}$ in every recursive invocation of these procedures is a proper subset of the argument $R \subseteq \mathrm{P}$ of the invoking procedure. It is also obvious that each individual invocation INV of the procedures uses at most polynomial space, when not counting the space used by the invocations that INV causes, so the total amount of memory used by these procedures can be indeed bounded by a polynomial. □

4. Distance-to-disabling functions (on general LTSs)

In this section we briefly look at the previous proof of Theorem 3 from a more general perspective. If we consider the problem of deciding bisimilarity on an LTS

$$\mathcal{L} = (S, \mathrm{ACT}, (\xrightarrow{a})_{a \in \mathrm{ACT}}),$$

we can think of various properties of two states $s_1, s_2 \in S$ that are necessary for them to be bisimilar. A simplest such property is that the sets of enabled actions must be the same for both s_1 and s_2; equivalently, the sets of disabled actions must be the same for both of them.

Let us call a *predicate* $P \subseteq S$ *bisimulation-invariant* if $s_1 \sim s_2$ entails that $s_1 \in P \iff s_2 \in P$. Instead of $s \in P$ we also write $P(s) = \mathit{true}$, or simply $P(s)$. More generally, a *function* $f : S \to \mathcal{X}$ (where $\mathcal{X}$ is any set, maybe a set of numbers) is *bisimulation-invariant* if $s_1 \sim s_2$ entails $f(s_1) = f(s_2)$. Hence for any set $\mathcal{G}$ of some bisimulation-invariant functions we have $\sim \subseteq \equiv_{\mathcal{G}}$, where

$$s_1 \equiv_{\mathcal{G}} s_2 \iff_{df} f(s_1) = f(s_2) \text{ for all } f \in \mathcal{G}.$$

We call such a *set* $\mathcal{G}$ *sound* if $\equiv_{\mathcal{G}} \subseteq \sim$. (From another viewpoint, such $\mathcal{G}$ could be also called "full" or "complete".)

In fact, in Theorem 3 we have implicitly used that a set of inductively defined "distance-to-disabling functions" constitutes an example of a sound set of bisimulation-invariant functions. This is formalized below, for general LTSs.

Definition 5. We assume an LTS $\mathcal{L} = (S, \text{Act}, (\xrightarrow{a})_{a\in\text{Act}})$.

- Given a predicate $P \subseteq S$, the *distance-to-P function* is $dist_P : S \to \mathbb{N}_\omega$ where $dist_P(s)$ is the length of a shortest $w \in \text{Act}^*$ such that $s \xrightarrow{w} s'$ and $P(s') = true$. If there is no such sequence, we put $dist_P(s) = \omega$ (in which case we have $dist_P(s') = \omega$ for all s' that are reachable from s).
- Let $d : S \to \mathbb{N}_\omega$ be $dist_P$ for some predicate P. The *change on d* caused by a transition $s \xrightarrow{a} s'$ is $\delta \in \mathbb{N}_{\omega,-1}$ where $\delta = d(s') - d(s)$. (By our conventions, $\delta = \omega$ iff $d(s') = \omega$, even if $d(s) = \omega$ as well).
- Given a tuple $\bar{d} = (d_1, d_2, \ldots, d_m)$ of functions $d_i = dist_{P_i}$, and a tuple $\bar{\delta} = (\delta_1, \delta_2, \ldots, \delta_m)$ of values $\delta_i \in \mathbb{N}_{\omega,-1}$, for $i = 1, 2, \ldots, m$, we define the predicate $\text{Disabled}_{(a,\bar{d},\bar{\delta})}$, for each $a \in \text{Act}$, as follows:

  $$\text{Disabled}_{(a,\bar{d},\bar{\delta})}(s) \Leftrightarrow_{df} d_i(s) < \omega \text{ for each } d_i \text{ in } \bar{d}, \text{ and there is no } s' \text{ such that}$$

  $$s \xrightarrow{a} s' \text{ and } d_i(s') - d_i(s) = \delta_i \text{ for all } i \in \{1, 2, \ldots, m\}.$$

  (Hence $\text{Disabled}_{(a,\bar{d},\bar{\delta})}(s) = false$ iff either $d_i(s) = \omega$ for some d_i in $\bar{d}$, or there is a transition $s \xrightarrow{a} s'$ for which $d_i(s') - d_i(s) = \delta_i$ for all $i \in \{1, 2, \ldots, m\}$.)
- The set $\text{DD}(\mathcal{L})$ of *distance-to-disabling functions on* $\mathcal{L}$, or of *DD-functions on* $\mathcal{L}$ for short, is the least set of functions of the type $S \to \mathbb{N}_\omega$ for which we have:

  for all $a \in \text{Act}$, $\bar{d} = (d_1, d_2, \ldots, d_m)$, and $\bar{\delta} = (\delta_1, \delta_2, \ldots, \delta_m)$, where $m \geq 0$, $d_i \in \text{DD}(\mathcal{L})$ and $\delta_i \in \mathbb{N}_{\omega,-1}$ for $i = 1, 2, \ldots, m$, the function $dist_{\text{Disabled}_{(a,\bar{d},\bar{\delta})}}$, denoted by $dd_{(a,\bar{d},\bar{\delta})}$, belongs to $\text{DD}(\mathcal{L})$.

We note that the predicate $\text{Disabled}_{(a,(),())}$ (where $m = 0$ and the tuples $\bar{d}$, $\bar{\delta}$ are empty) is true for the states $s \in S$ where a is disabled (i.e., there is no s' such that $s \xrightarrow{a} s'$). The functions $dd_{(a,(),())}$ can be also viewed as constituting a base case of an inductive definition of the set $\text{DD}(\mathcal{L})$.

Proposition 6. *Given an image-finite LTS* $\mathcal{L} = (S, \text{Act}, (\xrightarrow{a})_{a\in\text{Act}})$, *i.e. the set* $\{s' \mid s \xrightarrow{a} s'\}$ *is finite for each pair* $s \in S, a \in \text{Act}$, *the set* $DD(\mathcal{L})$ *is a sound set of bisimulation-invariant functions, hence*

$$s_1 \sim s_2 \text{ iff } s_1 \equiv_{DD(\mathcal{L})} s_2 \text{ (i.e., iff } d(s_1) = d(s_2) \text{ for all } d \in DD(\mathcal{L})).$$

Proof. *Soundness* ($\equiv_{\text{DD}(\mathcal{L})} \subseteq \sim$)

We show that $\equiv_{\text{DD}(\mathcal{L})}$ is a bisimulation. For the sake of contradiction, suppose that $s_1 \equiv_{\text{DD}(\mathcal{L})} s_2$, i.e., $d(s_1) = d(s_2)$ for all $d \in \text{DD}(\mathcal{L})$, and let $s_1 \xrightarrow{a} s_1'$ be such that for each transition $s_2 \xrightarrow{a} s_2'$ we have $s_1' \not\equiv_{\text{DD}(\mathcal{L})} s_2'$. There is some transition $s_2 \xrightarrow{a} s_2'$ since otherwise $dd_{(a,(),())}(s_1) > dd_{(a,(),())}(s_2) = 0$ (which contradicts with $s_1 \equiv_{\text{DD}(\mathcal{L})} s_2$). Since there are only finitely many s_2' such that $s_2 \xrightarrow{a} s_2'$ (since $\mathcal{L}$ is image-finite), there are finitely many $d_1, d_2, \ldots, d_m \in \text{DD}(\mathcal{L})$ such that $d_i(s_1) = d_i(s_2) < \omega$ for all $i \in \{1, 2, \ldots, m\}$ and for each such s_2' ($s_2 \xrightarrow{a} s_2'$) there is $i \in \{1, 2, \ldots, m\}$ for which $d_i(s_2') \neq d_i(s_1')$. We thus derive that $\text{Disabled}_{(a,\bar{d},\bar{\delta})}(s_1) = false$ and $\text{Disabled}_{(a,\bar{d},\bar{\delta})}(s_2) = true$ for $\bar{d} = (d_1, d_2, \ldots, d_m)$ and $\bar{\delta} = (\delta_1, \delta_2, \ldots, \delta_m)$ where $\delta_i = d_i(s_1') - d_i(s_1)$ (for all $i \in \{1, 2, \ldots, m\}$); but this entails that $dd_{(a,\bar{d},\bar{\delta})}(s_1) > dd_{(a,\bar{d},\bar{\delta})}(s_2) = 0$, which contradicts with $s_1 \equiv_{\text{DD}(\mathcal{L})} s_2$.

Completeness ($\sim \subseteq \equiv_{\text{DD}(\mathcal{L})}$); this part does not depend on the image-finiteness of $\mathcal{L}$.

We will show that $s_1 \sim s_2$ entails $d(s_1) = d(s_2)$ for all $d \in \text{DD}(\mathcal{L})$, by using the inductive definition of $\text{DD}(\mathcal{L})$. It thus suffices to show that $s_1 \sim s_2$ entails $d(s_1) = d(s_2)$ for

$$d = dist_{\text{Disabled}_{(a,\bar{d},\bar{\delta})}}$$

where $a \in \text{Act}$, $\bar{d} = (d_1, d_2, \ldots, d_m)$, and $\bar{\delta} = (\delta_1, \delta_2, \ldots, \delta_m)$, assuming (by the induction hypothesis) that $\sim \subseteq \equiv_{\mathcal{G}}$ for $\mathcal{G} = \{d_1, d_2, \cdots, d_m\}$.

For the sake of contradiction, we suppose that $s_1 \sim s_2$ and $d(s_1) > d(s_2)$. Then there is a shortest word $w \in \text{Act}^*$ such that $s_2 \xrightarrow{w} s_2'$ where $d(s_2') = 0$, which entails that $\text{Disabled}_{(a,\bar{d},\bar{\delta})}(s_2') = true$, and thus also $d_i(s_2') < \omega$ for all $i \in \{1, 2, \ldots, m\}$. Since $s_1 \sim s_2$, there is s_1' such that $s_1 \xrightarrow{w} s_1'$, $s_1' \sim s_2'$, and $d(s_1') \geq 1$; hence $\text{Disabled}_{(a,\bar{d},\bar{\delta})}(s_1') = false$ but $d_i(s_1') = d_i(s_2') < \omega$ for all $i \in \{1, 2, \ldots, m\}$ (since $\sim \subseteq \equiv_{\mathcal{G}}$).

We thus have $s'_1 \xrightarrow{a} s''_2$ where $d_i(s''_1) - d_i(s'_1) = \delta_i$ for all $i \in \{1, 2, \ldots, m\}$. Since $s'_1 \sim s'_2$, we have $s'_2 \xrightarrow{a} s''_2$ where $s''_1 \sim s''_2$; we thus cannot have that $\bar{d}$ and $\bar{\delta}$ are empty, since $\textsc{Disabled}_{(a,(),())}(s'_2)$ means that a is disabled in s'_2. Nevertheless, we necessarily have $d_i(s''_2) - d_i(s'_2) \neq \delta_i$ for some $i \in \{1, 2, \ldots, m\}$. We have thus got $s''_1 \sim s''_2$ where $d_i(s''_1) \neq d_i(s''_2)$ for some $i \in \{1, 2, \ldots, m\}$; this contradicts with (the induction hypothesis) $\sim \subseteq \equiv_{\mathcal{G}}$. □

We note that the LTSs $\mathcal{L}_N$ for BPP-nets N are image-finite, and the respective functions $d \in \mathsf{DD}(\mathcal{L}_N)$ correspond to the functions $\textsc{norm}_Q$ for certain sets Q of places; this also entails that the sets $\mathsf{DD}(\mathcal{L}_N)$ are finite, in fact. Some remarks about this issue are added in the next section.

5. Some remarks on related research and open problems

The author conjectured in [1] that the technique of distance-to-disabled functions, presented in the case of (strong) bisimilarity on the class BPP, could turn out useful also for showing decidability of *weak bisimilarity* on BPP. In this case we also consider *silent* transitions $p \xrightarrow{\tau} M$ where τ denotes a silent (i.e. internal) action of the systems. Then each move $M_1 \xrightarrow{a} M'_1$ can be matched by a "long" move $M_2 \stackrel{a}{\Longrightarrow} M'_2$, i.e. by a sequence

$$M_2 \xrightarrow{\tau} \cdots \xrightarrow{\tau} \xrightarrow{a} \xrightarrow{\tau} \cdots \xrightarrow{\tau} M'_2; \tag{9}$$

a move $M_1 \xrightarrow{\tau} M'_1$ can be also matched by $M_2 \xrightarrow{\varepsilon} M_2$. Nevertheless, the conjecture has not been confirmed so far, and the decidability of weak bisimilarity on BPP remains open. It is known to be semidecidable [18] but a crucial problem is that the respective LTSs, with transition-relations $\stackrel{a}{\Longrightarrow}$ (for $a \in \textsc{Act} \cup \{\tau\}$) are not image-finite (since there is no bound on τ-sequences, which can thus generate multisets of arbitrary sizes). For normed BPP, this problem was handled in [19] for a finer equivalence called *branching bisimilarity*, another established behavioural equivalence (see, e.g. [20]). Here the τ-moves in the matching sequence (9) are required to not change the equivalence-class.

The problem of decidability of branching bisimilarity was studied also by Yuxi Fu, but for the related model of normed BPA processes (where transitions are of the type $p \xrightarrow{a} \alpha$ where α is a finite sequence rather than a finite multiset). He answered this decidability question positively in [16], by introducing a nice notion of a *norm* that is *defined semantically*: instead of counting the smallest number of moves needed to reach the empty process, we count the smallest number of equivalence-class changes on paths to the empty process. Unlike the standard ("syntactic") norm, such a semantic norm is not clearly computable. Nevertheless, reasoning about this norm allowed Fu to derive a decision algorithm and proving its correctness. Fu's result initiated a research effort to clarify the complexity of this decidable problem. An original observation that EXPTIME-hardness should follow by Mayr's result for weak bisimilarity [21] turned out not so obvious, but was nicely confirmed in [22]. Regarding the upper bound, there are two papers published at the same conference, [23] and [24], and one journal paper [25]. We note that the decidability questions for unnormed versions, of BPP and BPA, are open for both weak bisimilarity and branching bisimilarity.

We finish with a note on (strong) bisimilarity on BPA. Unlike for BPP, the complexity is not fully clarified so far. The problem is known to be EXPTIME-hard [26] and in 2-EXPTIME (claimed in [27] and explicitly proven in [28]). Though in this case the respective labelled transition systems are image-finite, the set of distance-to-disabling functions for a given LTS is generally infinite, unlike the case of BPP that has been studied in this paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The author thanks the anonymous reviewers for their helpful comments.

References

[1] P. Jančar, Strong bisimilarity on Basic Parallel Processes is PSPACE-complete, in: Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science (LICS-03), IEEE Computer Society, 2003, pp. 218–227.
[2] J. Srba, Strong bisimilarity and regularity of Basic Parallel Processes is PSPACE-hard, in: Proceedings of STACS'02, in: Lecture Notes in Computer Science, vol. 2285, Springer-Verlag, 2002, pp. 535–546.
[3] J. Srba, Strong bisimilarity of simple process algebras: complexity lower bounds, Acta Inform. 39 (6–7) (2003) 469–499, https://doi.org/10.1007/s00236-003-0116-9.
[4] R. Milner, Communication and Concurrency, Prentice-Hall, 1989.
[5] R. Paige, R.E. Tarjan, Three partition refinement algorithms, SIAM J. Comput. 16 (6) (1987) 973–989.
[6] P.C. Kanellakis, S.A. Smolka, CCS expressions, finite state processes, and three problems of equivalence, Inf. Comput. 86 (1) (1990) 43–68.
[7] J.L. Balcázar, J. Gabarró, M. Santha, Deciding bisimilarity is P-complete, Form. Asp. Comput. 4 (6A) (1992) 638–648.

[8] J.C.M. Baeten, J.A. Bergstra, J.W. Klop, Decidability of bisimulation equivalence for processes generating context-free languages, J. ACM 40 (3) (1993) 653–682.
[9] O. Burkart, D. Caucal, F. Moller, B. Steffen, Verification on infinite structures, in: J. Bergstra, A. Ponse, S. Smolka (Eds.), Handbook of Process Algebra, Elsevier Science, 2001, pp. 545–623, Ch. 9.
[10] J. Srba, Roadmap of infinite results, in: Current Trends in Theoretical Computer Science, the Challenge of the New Century, Vol. 2: Formal Models and Semantics, World Scientific Publishing Co., 2004, pp. 337–350. See an updated version at http://people.cs.aau.dk/~srba/roadmap/.
[11] S. Christensen, Y. Hirshfeld, F. Moller, Bisimulation equivalence is decidable for Basic Parallel Processes, in: Proceedings of CONCUR '93, in: Lecture Notes in Computer Science, vol. 715, Springer-Verlag, 1993, pp. 143–157.
[12] Y. Hirshfeld, M. Jerrum, F. Moller, A polynomial-time algorithm for deciding bisimulation equivalence of normed Basic Parallel Processes, Math. Struct. Comput. Sci. 6 (3) (1996) 251–259.
[13] S. Fröschle, S. Lasota, Normed processes, unique decomposition, and complexity of bisimulation equivalences, Electron. Notes Theor. Comput. Sci. 239 (2009) 17–42, https://doi.org/10.1016/j.entcs.2009.05.028.
[14] P. Jančar, M. Kot, Bisimilarity on normed Basic Parallel Processes can be decided in time $O(n^3)$, in: Proceedings of the Third International Workshop on Automated Verification of Infinite-State Systems – AVIS 2004, 2004.
[15] M. Kot, Z. Sawa, Bisimulation equivalence of a BPP and a finite-state system can be decided in polynomial time, Electron. Notes Theor. Comput. Sci. 138 (3) (2005) 49–60, proceedings of Infinity 2004.
[16] Y. Fu, Checking equality and regularity for normed BPA with silent moves, in: Proceedings of ICALP 2013, Part II, in: Lecture Notes in Computer Science, vol. 7966, Springer, 2013, pp. 238–249.
[17] R. Mayr, Process rewrite systems, Inf. Comput. 156 (1) (2000) 264–286.
[18] J. Esparza, Petri nets, commutative context-free grammars, and Basic Parallel Processes, Fundam. Inform. 31 (1) (1997) 13–25.
[19] W. Czerwinski, P. Hofman, S. Lasota, Decidability of branching bisimulation on normed commutative context-free processes, Theory Comput. Syst. 55 (1) (2014) 136–169, https://doi.org/10.1007/s00224-013-9505-9.
[20] R.J. van Glabbeek, W.P. Weijland, Branching time and abstraction in bisimulation semantics, J. ACM 43 (3) (1996) 555–600, https://doi.org/10.1145/233551.233556.
[21] R. Mayr, Weak bisimilarity and regularity of context-free processes is exptime-hard, Theor. Comput. Sci. 330 (3) (2005) 553–575, https://doi.org/10.1016/j.tcs.2004.10.008.
[22] M. Huang, Q. Yin, Two lower bounds for BPA, in: Proceedings of CONCUR 2017, in: LIPIcs, vol. 85, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, pp. 20:1–20:16.
[23] W. Czerwinski, P. Jančar, Branching bisimilarity of normed BPA processes is in NEXPTIME, in: Proceedings of Symposium on Logic in Computer Science, LICS 2015, IEEE Computer Society, 2015, pp. 168–179.
[24] C. He, M. Huang, Branching bisimilarity on normed BPA is EXPTIME-complete, in: Proceedings of Symposium on Logic in Computer Science, LICS 2015, IEEE Computer Society, 2015, pp. 180–191.
[25] P. Jančar, Branching bisimilarity of normed BPA processes as a rational monoid, Logical Methods in Computer Science 13 (4), https://doi.org/10.23638/LMCS-13(4:17)2017.
[26] S. Kiefer, BPA bisimilarity is EXPTIME-hard, Inf. Process. Lett. 113 (4) (2013) 101–106, https://doi.org/10.1016/j.ipl.2012.12.004.
[27] O. Burkart, D. Caucal, B. Steffen, An elementary bisimulation decision procedure for arbitrary context-free processes, in: Proceedings of MFCS'95, in: Lecture Notes in Computer Science, vol. 969, Springer, 1995, pp. 423–433.
[28] P. Jančar, Bisimilarity on basic process algebra is in 2-ExpTime (an explicit proof), Log. Methods Comput. Sci. 9 (1) (2013), https://doi.org/10.2168/LMCS-9(1:10)2013.