Complexity of Deciding Bisimilarity between Normed BPA and Normed BPP[☆]

Petr Jančar, Martin Kot, Zdeněk Sawa*

Center for Applied Cybernetics,
Dept. of Computer Science, Technical University of Ostrava,
17. listopadu 15, Ostrava-Poruba, 708 33, Czech Republic

Abstract

We present a polynomial-time algorithm deciding bisimilarity between a normed BPA process and a normed BPP process, with running time $O(n^7)$. This improves the previously known exponential upper bound by Černá, Křetínský, Kučera (1999). We first suggest an $O(n^3)$ transformation of the BPP process into "prime form". Our algorithm then relies on a polynomial bound for a "finite-state core" of the transition system generated by the (transformed) BPP process.

Keywords: bisimulation equivalence, Basic Process Algebra, Basic Parallel Processes, complexity

[☆]The authors acknowledge the support by the Czech Ministry of Education, Grant No. 1M0567.

*Corresponding author — address:
Zdeněk Sawa
Dept. of Computer Science, Technical University of Ostrava,
17. listopadu 15, Ostrava-Poruba, 708 33, Czech Republic
e-mail: zdenek.sawa@vsb.cz
Tel.: +420 597 324 437, Fax: +420 597 323 099

Email addresses: petr.jancar@vsb.cz (Petr Jančar), martin.kot@vsb.cz (Martin Kot), zdenek.sawa@vsb.cz (Zdeněk Sawa)

1. Introduction

Decidability and complexity of bisimilarity on various classes of processes is a classical topic in process algebra and concurrency theory; see, e.g., [1, 2] for surveys.

One long-standing open problem is the decidability question for the class PA (process algebra), which comprises "context-free" rewrite systems using both sequential and parallel composition. For the subcase of *normed* PA, a procedure working in doubly-exponential nondeterministic time was shown by Hirshfeld and Jerrum [3].

More is known about the "sequential" subclass called BPA (Basic Process Algebra) and the "parallel" subclass called BPP (Basic Parallel Processes). In the case of BPA, the best known algorithm for deciding bisimilarity seems to have doubly-exponential upper bound [4, 1]; the problem is known to be PSPACE-hard [5]. In the case of BPP, the problem is PSPACE-complete [6, 7]. A polynomial-time algorithm for *normed* BPA was shown in [8] (with an upper bound $O(n^{13})$); more recently, an algorithm with running time in $O(n^8 polylog\ n)$ was shown in [9]. For normed BPP, a polynomial time algorithm was presented in [10] (without a precise complexity analysis), based on so called *prime decompositions*; the upper bound $O(n^3)$ was shown in [11] by another algorithm, based on so called *dd-functions* (distance-to-disabling functions).

The most difficult part of the above mentioned algorithm for normed PA [3] deals with the case when (a process expressed as) sequential composition is bisimilar to (a process expressed as) parallel composition. A basic subproblem is to analyze when a BPA process is bisimilar to a BPP process. Černá, Křetínský, Kučera [12] have shown that this subproblem is decidable in the *normed* case; their suggested algorithm is exponential. Decidability in the general (unnormed) case was shown in [13], without providing any complexity bound.

In this paper, we revisit the normed case, and we present a *polynomial* algorithm deciding whether a given normed BPA process α is bisimilar to a given normed BPP process M. An important ingredient is a new algorithm, based on *dd*-functions, which transforms the normed BPP process M into "prime form" where bisimilarity coincides with equality; time complexity of this transformation is $O(n^3)$. We note that such a transformation could be

based on prime decompositions from [10] but with worse complexity (which was, in fact, not analyzed in [10]). A further main idea is to derive a polynomial bound on a "finite-state core" of the transition system generated by the (transformed) BPP process M. If the size of the constructed finite-state core exceeds the derived bound, our decision algorithm answers negatively; otherwise it constructs a BPA process α' which is bisimilar to M, and the final step is to decide if the BPA processes α and α' are bisimilar.

To derive polynomiality, the mentioned final step can be handled by referring to [8] or [9]. To get a better complexity upper bound, namely $O(n^7)$, we suggest a simple self-contained algorithm, which exploits the fact that α' is "almost" a finite-state process.

As a side result, our approach also shows a clear polynomial time algorithm, with running time $O(n^3)$, testing if there exists a bisimilar BPA process to a given BPP process; polynomiality was shown in [12], with no bound on the polynomial degree. Another side result is an algorithm for deciding bisimilarity between a given BPA process and a given finite-state process, with running time $O(n^4)$. Polynomiality of this problem was already shown by Kučera and Mayr [14]. In fact, they provided an $O(n^{12})$ algorithm for the more general case of *weak* bisimilarity; the complexity for the special case of (strong) bisimilarity was not analyzed.

The paper is structured as follows. Section 2 contains basic definitions, and Section 3 describes the transformation of a normed BPP system into prime form. Section 4 provides a polynomial bound on the size of the finite-state core. Section 5 finishes the main polynomiality proof, and in Section 6 we develop a finer algorithm allowing to derive the upper bound $O(n^7)$.

A preliminary version of this paper (with no complexity analysis) appeared at Concur'08 [15].

2. Definitions

We use $\mathbb{N} = \{0, 1, 2, \ldots\}$ to denote the set of nonnegative integers, and we put $\mathbb{N}_{-1} = \mathbb{N} \cup \{-1\}$.

For a set X, $|X|$ denotes the size of X, X^+ denotes the set of nonempty sequences of elements of X, and $X^* = X^+ \cup \{\varepsilon\}$ where ε is the empty sequence. The length of a sequence $x \in X^*$ is denoted by $|x|$ ($|\varepsilon| = 0$). We

use x^k (where $x \in X^*$, $k \in \mathbb{N}$) to denote the sequence $xx \cdots x$ where x is repeated k times (in particular $x^0 = \varepsilon$).

A *labelled transition system* (LTS) is a triple $(S, \mathcal{A}, \longrightarrow)$, where S is a set of *states*, $\mathcal{A}$ is a finite set of *actions*, and $\longrightarrow \subseteq S \times \mathcal{A} \times S$ is a *transition relation*. We write $s \xrightarrow{a} s'$ instead of $(s, a, s') \in \longrightarrow$ and we extend this notation to elements of $\mathcal{A}^*$ in the natural way. We write $s \longrightarrow s'$ if there is $a \in \mathcal{A}$ such that $s \xrightarrow{a} s'$ and $s \longrightarrow^* s'$ if $s \xrightarrow{w} s'$ for some $w \in \mathcal{A}^*$. By $s \xrightarrow{w}$ we denote that there is some s' such that $s \xrightarrow{w} s'$.

Let $(S, \mathcal{A}, \longrightarrow)$ be an LTS. A binary relation $\mathcal{R} \subseteq S \times S$ is a *bisimulation* if for each $(s, t) \in \mathcal{R}$ and each $a \in \mathcal{A}$ we have:

- $\forall s' \in S : s \xrightarrow{a} s' \Rightarrow (\exists t' : t \xrightarrow{a} t' \wedge (s', t') \in \mathcal{R})$, and
- $\forall t' \in S : t \xrightarrow{a} t' \Rightarrow (\exists s' : s \xrightarrow{a} s' \wedge (s', t') \in \mathcal{R})$.

Informally we say that transition $s \xrightarrow{a} s'$ can be *matched* by a transition $t \xrightarrow{a} t'$ where $(s', t') \in \mathcal{R}$, and vice versa.

States s and t are *bisimulation equivalent (bisimilar)*, written $s \sim t$, if they are related by some bisimulation. We can also relate states of two different LTSs, by considering the disjoint union of these LTSs.

A *BPA system*, or *BPA* for short, can be viewed as a context-free grammar in Greibach normal form. Formally it is a triple $\Sigma = (V_\Sigma, \mathcal{A}_\Sigma, \Gamma_\Sigma)$, where V_Σ is a finite set of *variables* (nonterminals), $\mathcal{A}_\Sigma$ is a finite set of *actions* (terminals) and $\Gamma_\Sigma \subseteq V_\Sigma \times \mathcal{A}_\Sigma \times V_\Sigma^*$ is a finite set of *rewrite rules*. We often use $V, \mathcal{A}, \Gamma$ without subscripts when the underlying BPA is clear from context. We also write $X \xrightarrow{a} \alpha$ instead of $(X, a, \alpha) \in \Gamma$. A *BPA process* is a pair (α, Σ) where Σ is a BPA system and $\alpha \in V^*$; we write just α when Σ is clear from context. A BPA Σ gives rise to the LTS $\mathcal{S}_\Sigma = (V^*, \mathcal{A}, \longrightarrow)$ where $\longrightarrow$ is induced from the rewrite rules by the following (deduction) rule: if $X \xrightarrow{a} \alpha$ then $X\beta \xrightarrow{a} \alpha\beta$ for every $\beta \in V^*$.

A *BPP system*, or *BPP* for short, is defined in a similar way, as a triple $\Delta = (V_\Delta, \mathcal{A}_\Delta, \Gamma_\Delta)$. The only difference is the deduction rule for the associated LTS $\mathcal{S}_\Delta$: if $X \xrightarrow{a} \alpha$ then $\gamma X \delta \xrightarrow{a} \gamma\alpha\delta$ for any $\gamma, \delta \in V^*$ (thus *any* occurrence of a variable can be rewritten, not just the first one). It is easy to observe that BPP processes α, β with the same Parikh image (i.e., containing the same number of occurrences of each variable) are bisimilar.

Hence BPP processes can be read modulo commutativity of concatenation and interpreted as multisets of variables; in the rest of the paper we interpret BPP processes in this way whenever convenient. This also suggests to identify a BPP system Δ with a *BPP net*, a labelled Petri net in which each place corresponds to a variable and each transition corresponds to a rewrite rule (and thus has a unique input place); we will freely do this in our later considerations.

Formally, a *BPP net* is a tuple $\Delta = (P_\Delta, Tr_\Delta, \mathrm{PRE}_\Delta, F_\Delta, \mathcal{A}_\Delta, l_\Delta)$ where P_Δ is a finite set of *places* (variables), Tr_Δ is a finite set of *transitions*, $\mathrm{PRE}_\Delta : Tr_\Delta \to P_\Delta$ is a function assigning an input place to each transition, $F_\Delta : (Tr_\Delta \times P_\Delta) \to \mathbb{N}$ is a *flow function*, $\mathcal{A}_\Delta$ is a finite set of *actions*, and $l_\Delta : Tr_\Delta \to \mathcal{A}_\Delta$ is a *labelling function*. We will use $P, Tr, \mathrm{PRE}, F, \mathcal{A}, l$ if the underlying BPP net is clear from context. We note that a transition $t \in Tr$ can be viewed as the rewrite rule $p \stackrel{a}{\longrightarrow} \alpha$ where $\mathrm{PRE}(t) = p$ and $F(t, p')$ is the number of occurrences of p' in α, for each $p' \in P$.

A BPP process is thus, in fact, a *marking*, i.e. a function $M : P \to \mathbb{N}$ which associates a finite number of *tokens* to each place. Thus p^k represents the marking M where all k tokens are in one place p ($M(p) = k$ and $M(p') = 0$ for each $p' \neq p$); $p^0 = \varepsilon$ represents the *zero marking* ($M(p) = 0$ for all $p \in P$).

A transition t is *enabled* at marking M if $M(\mathrm{PRE}(t)) \geq 1$. An enabled transition t may fire from M, producing a marking M' defined by

$$M'(p) = \begin{cases} M(p) - 1 + F(t,p) & \text{if } p = \mathrm{PRE}(t) \\ M(p) + F(t,p) & \text{otherwise} \end{cases} .$$

This is denoted by $M \stackrel{t}{\longrightarrow} M'$; the notation is extended to $M \stackrel{\sigma}{\longrightarrow} M'$ for sequences $\sigma \in T^*$. We write $M \stackrel{\sigma}{\longrightarrow}$ if $M \stackrel{\sigma}{\longrightarrow} M'$ for some M'.

In the above sense, a BPP Δ gives rise to the LTS $\mathcal{S}_\Delta = (\mathcal{M}_\Delta, \mathcal{A}, \longrightarrow)$ where $\mathcal{M}_\Delta = \mathbb{N}^P$ is the set of all markings (of the respective BPP net), and $M \stackrel{a}{\longrightarrow} M'$ iff there is some $t \in Tr$ such that $l(t) = a$ and $M \stackrel{t}{\longrightarrow} M'$.

In the rest of the paper we use symbols $\alpha, \beta, \ldots$ for both BPA processes and BPP processes, and $M_1, M_2, \ldots$ only for the latter.

We say that a BPA system Σ (a BPP net Δ) is *normed* iff $\alpha \longrightarrow^* \varepsilon$ for each state α of $\mathcal{S}_\Sigma$ ($\mathcal{S}_\Delta$). We use nBPA (nBPP) for normed BPA (normed BPP).

Our central problem, denoted nBPA-nBPP-BISIM, is defined as follows:

INSTANCE: A normed BPA-process (α_0, Σ), a normed BPP-process (M_0, Δ).

QUESTION: Is $\alpha_0 \sim M_0$ (in the disjoint union of $\mathcal{S}_\Sigma$ and $\mathcal{S}_\Delta$)?

As the size n of an instance of nBPA-nBPP-BISIM we understand the number of bits needed for its (natural) presentation; in particular we consider the numbers $F(t,p)$ in Δ and the numbers in M_0 to be written in binary.

In the rest of this section we assume a fixed nBPA Σ and a fixed nBPP Δ. By a *state* we generally mean a state in the disjoint union of $\mathcal{S}_\Sigma$ and $\mathcal{S}_\Delta$.

Let α be a state (of $\mathcal{S}_\Sigma$ or $\mathcal{S}_\Delta$). The *norm* of α, denoted $\|\alpha\|$, is the length of the shortest $w \in \mathcal{A}^*$ such that $\alpha \stackrel{w}{\longrightarrow} \varepsilon$. Note that this also defines norm $\|X\|$ for each variable (place) X. We now note some obvious properties of norms.

- If $\alpha \neq \varepsilon$ then $\|\alpha\| > 0$ for any state α.
- In each nBPA (or nBPP), there is at least one variable (place) with norm 1.
- If a rule $X \stackrel{a}{\longrightarrow} \alpha$ is used in a transition $\beta \stackrel{a}{\longrightarrow} \beta'$ then $\|\beta'\| - \|\beta\| = \|\alpha\| - \|X\|$.
- $\|\alpha\beta\| = \|\alpha\| + \|\beta\|$ (for the BPP-net representation it means $\|M_1 + M_2\| = \|M_1\| + \|M_2\|$ where the sum $M_1 + M_2$ is defined componentwise).
- If $\alpha \sim \beta$ then $\|\alpha\| = \|\beta\|$.

Note also that if $\alpha_1 \sim \alpha_2$, $w \in \mathcal{A}^*$ and $\alpha_1 \stackrel{w}{\longrightarrow} \alpha_1'$ then there must be a *matching sequence* $\alpha_2 \stackrel{w}{\longrightarrow} \alpha_2'$ such that $\alpha_1' \sim \alpha_2'$ (and thus also $\|\alpha_1'\| = \|\alpha_2'\|$).

We will later use the following straightforward proposition.

Proposition 1. *The norms $\|X\|$, $\|p\|$ for $X \in V_\Sigma$, $p \in P_\Delta$ can be written in $O(n)$ bits, thus all of them together in $O(n^2)$ bits. All these norms can be computed in time $O(n^3)$.*

For two states α_1, α_2 we write $\alpha_1 \longrightarrow_R \alpha_2$ if $\alpha_1 \longrightarrow \alpha_2$ and $\|\alpha_2\| = \|\alpha_1\| - 1$. Such a step is called a *norm-reducing step* and the respective rule (transition) is also called *norm reducing*. We write $\alpha_1 \longrightarrow_R^* \alpha_2$ if there is a sequence

(called *norm reducing sequence*) of norm reducing steps leading from α_1 to α_2. For each variable (place) X there is at least one norm-reducing rule (transition) $X \longrightarrow_R \alpha$.

We finish by a few notions concerning the BPP net Δ.

For a marking M and a set $Q \subseteq P$ we define $\|M\|_Q$, the *norm of M wrt Q*, as the length of the shortest $w \in \mathcal{A}^*$ such that $M \stackrel{w}{\longrightarrow} M'$ where $M'(p) = 0$ for all $p \in Q$. In fact, $\|M\|_Q = \sum_{p \in Q} c_p \cdot M(p)$ where $c_p = \|p\|_Q$.

It is easy to derive the following useful fact.

Proposition 2. *For every $Q \subseteq P$ and $t \in Tr$ there is $\delta \in \mathbb{N}_{-1}$ such that $M \stackrel{t}{\longrightarrow} M'$ implies $\|M'\|_Q = \|M\|_Q + \delta$ (for all M, M').*

A place $p \in P$ is *unbounded* in (M_0, Δ) iff for each $c \in \mathbb{N}$ there is a marking M' such that $M_0 \longrightarrow^* M'$ and $M'(p) > c$.

We define $Tok(M) = \sum_{p \in P} M(p)$ and $Car(M) = \{p \in P \mid M(p) \geq 1\}$.

A place p is called a *single final place*, an SF-place, if all transitions that take a token from p are of the form $p \stackrel{a}{\longrightarrow} p^k$, $k \geq 0$ (they can only put tokens back to p). It is easy to see that $\|p\| = 1$ for every SF-place p (since Δ is normed). We say that p is a non-SF-place if it is not an SF-place.

3. Normed BPP systems in prime form

We say that a *BPP net* Δ is *in prime form* if bisimilarity coincides with identity on the generated LTS, i.e., $M \sim M'$ iff $M = M'$. (In this case, each place p is a "prime" since it is not equivalent to a composition of other places.) Prime form is technically convenient for developing our main algorithm; this section shows a relevant transformation (Theorem 9).

It follows from the unique decomposition results in [10] that for each normed BPP system Δ there is an equivalent normed BPP system Δ' in prime form, and that Δ' can be constructed from Δ in polynomial time using the algorithm, described in [10], which computes certain prime decompositions of BPP-variables (i.e., BPP-net places); it is a polynomial time algorithm but its precise complexity has not been analyzed. We proceed in another way, based on the *dd*-functions, which yields a transformation with time complexity $O(n^3)$.

The main idea can be sketched as follows. Given a normed BPP system $\Delta = (P, Tr, \text{PRE}, F, \mathcal{A}, l)$, let $T_a \subseteq Tr$ be the set of transitions with label $a \in \mathcal{A}$. It is clear that $M \sim M'$ implies that the distance to disabling T_a is the same in both M and M'; by this distance in M we mean the length of the shortest w such that $M \xrightarrow{w} M_1$ and all $t \in T_a$ are disabled in M_1. In other words, we must have $\|M\|_{\text{PRE}(T_a)} = \|M'\|_{\text{PRE}(T_a)}$ when $M \sim M'$. ($\text{PRE}(T) = \{\text{PRE}(t) \mid t \in T\}$.) Now suppose, e.g., that $T \subseteq T_a$ consists of all transitions with label a such that performing any $t \in T$ changes the norm wrt $\text{PRE}(T_a)$ by $+3$ and the norm wrt $\text{PRE}(T_b)$ by -1, for some $b \in \mathcal{A}$ ($M_1 \xrightarrow{t} M_2$ implies $\|M_2\|_{\text{PRE}(T_a)} = \|M_1\|_{\text{PRE}(T_a)} + 3$ and $\|M_2\|_{\text{PRE}(T_b)} = \|M_1\|_{\text{PRE}(T_b)} - 1$). Then $M \sim M'$ necessarily implies $\|M\|_Q = \|M'\|_Q$ for $Q = \text{PRE}(T)$. These observations have been refined in [6] to devise an algorithm for general BPP, which was then instantiated to normed BPP in [11].

Given a normed BPP system $\Delta = (P, Tr, \text{PRE}, F, \mathcal{A}, l)$, of size n, the algorithm from [11] finishes in time $O(n^3)$ and constructs a partition

$$\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$$

of the set Tr of transitions; denoting $d_i(M) = \|M\|_{\text{PRE}(T_i)}$, it holds that

$$M \sim M' \text{ iff } d_i(M) = d_i(M') \text{ for all } i = 1, 2, \ldots, m\,.$$

Moreover, each class T_i is characterized by its unique pair (a_i, δ_i) where a_i is the label of all $t \in T_i$ and

$$\delta_i = (\delta_{i1}, \delta_{i2}, \ldots, \delta_{im})$$

is the vector in $(\mathbb{N}_{-1})^m$ capturing the following change, for any M, M':

$$\text{if } M \xrightarrow{t} M' \text{ for } t \in T_i \text{ then } d(M') = d(M) + \delta_i$$

where $d(M)$ denotes the vector $(d_1(M), d_2(M), \ldots, d_m(M))$. For convenience, we say *transition (of the type)* t_i when meaning any transition $t \in T_i$.

Similarly as Proposition 1, we can derive the following fact (proven in detail in [11]).

Proposition 3. *Each δ_{ij} can be written in space $O(n)$, and thus all pairs (a_i, δ_i) together in space $O(n^3)$.*

Due to the normedness, for every class T_i ($i \in \{1, 2, \ldots, m\}$) there is at least one transition t_j ($j \in \{1, 2, \ldots, m\}$) which decreases d_i (when t_j is enabled in M, which also entails $d_i(M) > 0$); this is concisely captured by the next proposition.

Proposition 4. $\forall i \exists j : \delta_{ji} = -1$.

We say that t_i is a *key transition* if it decreases some component of d, i.e. some d_j. Formally we define

$$\mathrm{KEY} = \{i \mid \delta_{ij} = -1 \text{ for some } j\}\,.$$

Proposition 5. $\forall i \in \mathrm{KEY} : \delta_{ii} = -1$.

Proof. If t_i (an element of T_i) decreases some d_j then for each M there is the greatest ℓ such that $M \xrightarrow{(t_i)^\ell}$. The last firing of t_i necessarily decreases d_i. Hence $\delta_{ii} = -1$. □

Thus for each $i \in \mathrm{KEY}$, $d_i(M)$ is the greatest ℓ such that $M \xrightarrow{(t_i)^\ell}$. (A shortest way to disable transitions in T_i is to fire them as long as possible.)

We say that t_i *reduces* t_j iff $\delta_{ij} = -1$. Formally we define the following relation RED on KEY:

$$\text{for } i, j \in \mathrm{KEY} \text{ we put } i \text{ RED } j \text{ iff } \delta_{ij} = -1\,.$$

Proposition 6. RED *is an equivalence relation.*

Proof. Reflexivity follows from Proposition 5.

To show symmetry, assume $i, j \in \mathrm{KEY}$ (so $\delta_{ii} = \delta_{jj} = -1$) such that $\delta_{ij} = -1$ but $\delta_{ji} \geq 0$ (for the sake of contradiction). Then firing t_j from M with $d_i(M) > 0$ as long as possible results in M' with $d_j(M') = 0$ and $d_i(M') > 0$. Thus $M' \xrightarrow{t_i}$, which is a contradiction since d_j can not be decreased.

Transitivity follows similarly: Suppose i RED j and j RED k but $\neg(i \text{ RED } k)$. So all $\delta_{ii}, \delta_{jj}, \delta_{kk}, \delta_{ij}, \delta_{ji}, \delta_{jk}, \delta_{kj}$ are -1 but $\delta_{ik} \geq 0$. Starting from M with $d_k(M) > 0$, we fire t_i as long as possible and thus get M' with $d_i(M') = d_j(M') = 0$ and $d_k(M') > 0$. Thus $M' \xrightarrow{t_k}$, which is a contradiction since d_j can not be decreased. □

The following two propositions will help us later to show the size of the constructed BPP in prime form equivalent to a given one. To simplify the notation, we put $Q_i = \text{PRE}(T_i)$ and note that $d_i(M) = \|M\|_{Q_i}$.

Proposition 7. *There are at most $|P|$ classes of equivalence* RED.

Proof. Let $T_N \subseteq Tr$ be some set of norm reducing transitions such that for each $p \in P$ there is exactly one $t \in T_N$ with $\text{PRE}(t) = p$ (i.e. $|T_N| = |P|$). It is thus sufficient to show that for each class C of RED there exists $i \in C$ and $t \in T_N \cap T_i$. Since the net can be emptied by using only the transitions from T_N, for each $i \in \text{KEY}$ there is $t \in T_N$ which decreases the norm wrt Q_i; thus $t = t_j$ for some $j \in \text{KEY}$. Hence j RED i, and therefore j belongs to the class of i. □

Proposition 8. *Let T_z be a class of the partition $\mathcal{T}$ containing non-key transitions. The number of classes C of equivalence* RED *such that t_i decreases d_z for some $i \in C$ is at most $|T_z|$.*

Proof. Let $T_{k_1}, T_{k_2}, \ldots, T_{k_x}$ be all classes of the partition $\mathcal{T}$ such that t_{k_i} decreases d_z. Let $T_K = T_{k_1} \cup \ldots \cup T_{k_x}$ and $Q_K = Q_{k_1} \cup \ldots \cup Q_{k_x}$. Since the transitions from T_K have to be able to decrease d_z to 0 (to empty the set Q_z), it holds $Q_z \subseteq Q_K$. Each transition from T_{k_i} reduces d_z, and so its input place is from Q_z. It follows that $Q_{k_i} \subseteq Q_z$ for each k_i, and so $Q_K \subseteq Q_z$. Therefore $Q_K = Q_z$ and thus $|Q_K| \leq |T_z|$.

To complete the proof, we need to show that the number of classes of RED containing some k_i is at most $|Q_K|$. The idea is similar as in the proof of Proposition 7. We can take some set $T_N \subseteq T_K$ such that for each $p \in Q_K$ there is exactly one transition $t \in T_N$ for which $\text{PRE}(t) = p$. Note that each $t \in T_N$ reduces d_z and $|T_N| = |Q_K|$. Using only the transitions from T_N, the set Q_K can be emptied and all $d_{k_1}, d_{k_2}, \ldots, d_{k_x}$ set to 0. For each $i \in \{k_1, k_2, \ldots, k_x\}$ there is $t \in T_N$ which decreases the norm wrt Q_i. It follows from the definition of T_N that $t = t_j$ for some $j \in \{k_1, k_2, \ldots, k_x\}$. Hence j RED i, and therefore j belongs to the class of i. □

Theorem 9. *There is an algorithm, with time complexity $O(n^3)$, which transforms a given normed BPP system $\Delta = (P, Tr, \text{PRE}, F, \mathcal{A}, l)$ into $\Delta' = (P', Tr', \text{PRE}', F', \mathcal{A}, l')$ in prime form, and any given state (marking) M of Δ into M' of Δ' such that $M \sim M'$. Moreover, $|Tr'| \leq |Tr|, |P'| \leq |P|$, and Δ' is represented in space $O(n^3)$.*

Proof. In the first phase we compute the partition $\mathcal{T} = \{T_1, T_2, \ldots, T_m\}$ as discussed above. We easily verify that $Q_i = Q_j$ for $i, j \in \mathrm{KEY}$ iff i RED j (and so j RED i).

The crucial idea is that Δ' will have a place p_C for each class C of the equivalence RED. For any M of Δ, the number $M'(p_C)$ will be equal to $\|M\|_{Q_i}$ for each $i \in C$. Proposition 7 implies $|P'| \leq |P|$.

For every $i \in \mathrm{KEY}$, we add a transition t'_i in Δ' such that $\mathrm{PRE}(t'_i) = p_C$ where $i \in C$; t'_i is labelled with a_i and it realizes the (nonnegative) change on the other places $p_{C'}$ according to δ_i (restricted to KEY). The number of transitions of Δ' added in this step is at most equal to the number of key transitions of Δ.

A non-key transition t_i (with $\delta_i \geq (0, 0, \ldots, 0)$) is enabled precisely when a (key) transition decreasing d_i is enabled (recall Proposition 4). Thus for each p_C where C contains j with $\delta_{ji} = -1$ we add a transition t with label a_i and $\mathrm{PRE}(t) = p_C$ which (gives a token back to p_C and) realizes the change δ_i (restricted to KEY). Proposition 8 implies that at most $|T_i|$ transitions are added to Δ' for every class T_i of non-key transitions.

A transition t can possibly increase all d_i. Therefore, an equivalent transition t' can have $|P'|$ output edges. The multiplicity of each output edge can be written in space $O(n)$ (recall Proposition 3).

Summing up, $\Delta' = (P', Tr', \mathrm{PRE}', F', \mathcal{A}, l')$ can be constructed in time $O(n^3)$ and represented in space $O(n^3)$. The correctness of the construction is obvious. □

In the following text we only consider BPP systems in prime form, if not stated otherwise.

4. A bound on the number of "not-all-in-one-SF" markings

In this section we prove the following theorem.

Theorem 10. *Assume a normed BPA system Σ, with the set V of variables, and a normed BPP system Δ in prime form, with the set P of places. The number of markings M of Δ such that $\alpha \sim M$ for some $\alpha \in V^+$ and M does not have all tokens in one SF-place is at most $4y^2$, where $y = \max\{|V|, |P|\}$.*

We start with a simple observation and then we bound the total number of tokens in the markings mentioned in the theorem.

Proposition 11. *If* $A\alpha \sim M$ *where* $\alpha \in V^*$ *and* $|Car(M)| \geq 2$ *then* $\|A\| \geq 2$.

Proof. From M with $|Car(M)| \geq 2$ we can obviously perform two different norm-reducing steps resulting in two different, and thus nonbisimilar, markings. On the other hand, any $A\alpha$ with $\|A\| = 1$ has a single outcome (namely α) of any norm-reducing step. □

Proposition 12. *If* $|Car(M)| \geq 2$ *and* $\alpha \sim M$ *for* $\alpha \in V^+$ *then* $Tok(M) \leq |V|$.

Proof. In fact, we prove a stronger proposition. To this aim, we order the variables from V into a sequence $A_1, A_2, \ldots, A_{|V|}$ so that $\|A_i\| \leq \|A_j\|$ for $i \leq j$. We now show the following claim: if $A_i\alpha \sim M$, where $|Car(M)| \geq 2$ (and $\alpha \in V^*$), then $Tok(M) \leq i$.

For the sake of contradiction, suppose a counterexample $A_i\alpha \sim M$, $Tok(M) \geq i{+}1$, for minimal i. Proposition 11 shows that $\|A_i\| \geq 2$, hence also $i \geq 2$ (since necessarily $\|A_1\| = 1$); therefore $Tok(M) \geq i{+}1 \geq 3$. There are two possible cases — $Car(M) = 2$ or $Car(M) \geq 3$. In the first case, at least one of the two marked places contains at least two tokens and so it can not be emptied in one step by a norm-reducing transition taking a token from this place, and it is obvious that the other marked place also remains marked after this step. In the second case, a norm-reducing step from an arbitrary marked place leads to a marking where at least two originally marked places remain marked. Hence there is at least one possible norm-reducing step $M \longrightarrow_R M'$ such that $|Car(M')| \geq 2$, $Tok(M') \geq i$. This step is matched by $A_i\alpha \longrightarrow_R A_j\beta\alpha$, $A_j\beta\alpha \sim M'$, where necessarily $\|A_j\| < \|A_i\|$ and thus $j < i$. This contradicts the minimality of our counterexample. □

From the definition of a non-SF-place follows that a token from any such place may be moved (not necessarily by a norm-reducing step) to another place in such a way that the total number of tokens is not decreased by this step. From this fact and from the previous proposition, we get the following corollary.

Corollary 13. *If* $\alpha \sim M$ *then* $M(p) \leq |V|$ *for every non-SF-place* p.

We now partition the markings in the theorem into four classes:

Class 1. Markings M with all tokens in one (non-SF) place ($|Car(M)| = 1$).

Class 2. Markings M with $|Car(M)| \geq 2$ where at least two different places with norm 1 are reachable; this necessarily means $M \longrightarrow^* M'$ for some M' satisfying $M'(p_1) \geq 1$, $M'(p_2) \geq 1$ for some $p_1 \neq p_2$ and $\|p_1\| = \|p_2\| = 1$.

Class 3. Markings M with $|Car(M)| \geq 2$ and with exactly one reachable ("sink") place p with norm 1, where p is a non-SF-place.

Class 4. Markings M with $|Car(M)| \geq 2$ and with exactly one reachable ("sink") place p with norm 1, where p is an SF-place.

We will show that each class contains at most y^2 markings by which we prove the theorem. (In fact, our bound is a bit generous, allowing to avoid some technicalities.)

Proposition 14. *The number of markings in Class 1 is bounded by* $|V| \cdot |P| \leq y^2$.

Proof. According to Corollary 13 there can be at most $|V|$ tokens in any non-SF-place and there are at most $|P|$ non-SF-places. It follows that Class 1 contains at most $|V| \cdot |P| \leq y^2$ markings. □

Proposition 15. *If* $\alpha \sim M$ *for* M *from Class 2 then* $\alpha = A$ *for some* $A \in V$. *Thus the number of markings in Class 2 is at most* $|V| \leq y$.

Proof. For the sake of contradiction, suppose $A\alpha \sim M$ where $\alpha \in V^+$ and M is from Class 2. We take a counterexample with the minimal length ℓ of a sequence v such that $M \xrightarrow{v} M'$ where $M'(p_1) \geq 1$, $M'(p_2) \geq 1$ for two different p_1, p_2 with norm 1. We note that $\|A\| \geq 2$ by Proposition 11, and first suppose $\ell > 0$. It is easy to verify that there is a move $M \longrightarrow M''$, matched by $A\alpha \longrightarrow B\beta\alpha$, $B\beta\alpha \sim M''$, where $|Car(M'')| \geq 2$ and the respective length ℓ decreased; this would be a contradiction with the assumed minimality. Thus $\ell = 0$, which means $M(p_1) \geq 1$, $M(p_2) \geq 1$. But then M certainly allows $M \longrightarrow^*_R M_1$, $M \longrightarrow^*_R M_2$ where $\|M_1\| = \|M_2\| = \|\alpha\| \geq 1$ and $M_1 \neq M_2$, and thus $M_1 \not\sim M_2$. On the other hand, $A\alpha$ can offer only α as the result of matching such sequences; hence $A\alpha \not\sim M$. □

Proposition 16. *If $A\alpha \sim M$ for $\alpha \in V^+$ and M from Class 3 or 4 then $M \longrightarrow_R^* p^{\|\alpha\|}$ where p is the sink place. Thus $\alpha \sim p^{\|\alpha\|}$.*

Proof. We prove the claim by induction on the norm $\|A\|$. Suppose $A\alpha \sim M$ as in the statement. Proposition 11 implies $\|A\| \geq 2$. Let p' be a place with the minimal norm from all places with norm greater than 1 marked in M (from the definitions of classes 3 and 4 follows that there is such place). Performing a norm-reducing transition with a token from p' corresponds to some $M \longrightarrow_R M'$, and this must be matched by $A\alpha \longrightarrow_R B\beta\alpha$, $B\beta\alpha \sim M'$, where $\|B\| < \|A\|$. If $Car(M) = \{p, p'\}$ and $\|p'\| = 2$ then $Car(M') = \{p\}$ and necessarily $M' = p^{\|B\beta\alpha\|}$. In all other cases $|Car(M')| \geq 2$ and $M' \longrightarrow_R^* p^{\|\beta\alpha\|}$ due to the induction hypothesis. Since obviously $p^{\|B\beta\alpha\|} \longrightarrow_R^* p^{\|\beta\alpha\|} \longrightarrow_R^* p^{\|\alpha\|}$, we are done. □

Proposition 17. *If $A\alpha \sim M$ where $\|\alpha\| \geq 2$ then M is not from Class 3.*

Proof. For the sake of contradiction, suppose $A\alpha \sim M$ with minimal possible $\|A\|$ such that $\|\alpha\| \geq 2$ and M is from Class 3, i.e. M has exactly one reachable sink place p which is a non-SF-place. Note that $\|A\| \geq 2$ by Proposition 11.

If there was a step $M \longrightarrow_R M'$ with $|Car(M')| \geq 2$, the matching $A\alpha \longrightarrow_R B\beta\alpha$ would lead to a contradiction with minimality of $\|A\|$. Since $|Car(M)| \geq 2$, the only remaining possibility is the following: $Tok(M) = 2$, $M(p) = 1$ and $M(p') = 1$ where $p' \longrightarrow_R p^k$ for $k = \|A\| + \|\alpha\| - 2 \geq 2$.

Since the sink place p is a non-SF-place, it must be in a cycle C with at least two places. Moving a token along C cannot generate new tokens, due to Corollary 13, so p' is not in C. On the other hand, C contains some p'' with $\|p''\| = 2$. Starting in M, we can move the token from p to p'', the norm being greater than $\|M\| = \|A\alpha\|$ along the way. For the resulting M' we obviously have $M' \longrightarrow_R^* M''$ for M'' satisfying $M''(p'') = 1$ and $\|M''\| = \|\alpha\|$. $A\alpha$ can match this only by reaching α but $\alpha \sim p^{\|\alpha\|}$ according to Proposition 16 and thus $\alpha \not\sim M''$. □

We can thus have $A\alpha \sim M$ for M from Class 3 only when $\|\alpha\| \leq 1$, and it is thus easy to derive the following corollary.

Corollary 18. *The number of markings in Class 3 is at most $|V|^2 \leq y^2$.*

Proposition 19. *The number of markings in Class 4 is at most* $|V|\cdot|P| \leq y^2$.

Proof. Let $A\alpha \sim M$ for M from Class 4, p being the respective SF-sink place. Using Proposition 16, we derive $\alpha \sim I^k$ where $k = \|\alpha\|$ and $I \in V$, $I \sim p$ (such I must exist since $M \longrightarrow^* p$). Thus $AI^k \sim M$ but $AI^k \not\sim I^m$ for any m since $I^m \sim p^m$ and $p^m \not\sim M$ (note that $p^m \neq M$ and Δ is in prime form).

Since $M \longrightarrow^*_R p^m$ for some m, there must be a (shortest) norm-reducing sequence $A \stackrel{w}{\longrightarrow} B\beta$ where $\beta \sim I^{\|\beta\|}$, $B \not\sim I^{\|B\|}$ but all norm-reducing transitions $B \stackrel{a}{\longrightarrow} \gamma$ satisfy $\gamma \sim I^{\|\gamma\|}$. The sequence $A\alpha \stackrel{w}{\longrightarrow} B\beta\alpha$ (where $B\beta\alpha \sim BI^{\|\beta\alpha\|}$) must be matched by some $M \stackrel{v}{\longrightarrow} M'$ where M' does not have all tokens in p but every norm-reducing transition from M' results in M'' with all tokens in p; it follows that M' has a single token (so we have at most $|P|$ possibilities for M').

This easily implies that there are at most $|V| \cdot |P| \leq y^2$ markings in Class 4. □

5. Problem nBPA-nBPP-BISIM is in PTIME

In this section we describe a polynomial time algorithm for nBPA-nBPP-BISIM.

In Subsection 5.1 we specify conditions, which a normed BPP process (M_0, Δ) satisfies iff there exists some normed BPA process (α_0, Σ) such that $\alpha_0 \sim M_0$. The conditions can be easily checked in a time polynomial with respect to the size of (M_0, Δ). If (M_0, Δ) satisfies them, such (α_0, Σ) can be constructed but its size can be exponential with respect to the size of (M_0, Δ).

A basic idea of an algorithm for nBPA-nBPP-BISIM is to construct an nBPA process bisimilar to a given nBPP process (if it exists) and then to use some (polynomial time) algorithm for deciding if this constructed nBPA process is bisimilar to the nBPA process from the instance of nBPA-nBPP-BISIM. The complexity of such algorithm would be exponential in general, but in Subsection 5.2 we show how results from Section 4 can be applied to obtain a polynomial time algorithm.

5.1. Deciding if there exists an nBPA process bisimilar to a given nBPP process

We start with some technical notions concerning unbounded places that will be useful for the characterization of an nBPP process, for which a bisimilar nBPA process exists.

We first note that if moving a token along a cycle C in a BPP system Δ generates new tokens in a place p and C is reachable (markable) from M_0 then p is *primarily unbounded* (in M_0). Any place which is unbounded is either primarily unbounded, or *secondarily unbounded*, which means reachable from a primarily unbounded place. Thus any unbounded place has at least one corresponding *pumping cycle*.

We say that an SF-place p is *growing* if there is a transition $p \xrightarrow{a} p^k$ for $k \geq 2$.

Lemma 20. *For (M_0, Δ), Δ being a normed BPP in prime form, there exists a normed BPA process (α_0, Σ) such that $\alpha_0 \sim M_0$, iff the following conditions hold:*

1. *each non-SF-place is bounded,*
2. *there is no M such that $M_0 \longrightarrow^* M$, $|Car(M)| \geq 2$ and $M(p) \geq 1$ for some growing SF-place p,*
3. *each non-growing SF-place p is bounded.*

Proof. ($\Rightarrow$) If 1. is violated then we cannot have $\alpha_0 \sim M_0$ (for any Σ with a finite variable set V) due to Corollary 13. If 2. or 3. is violated then, for any $c \in \mathbb{N}$, $M_0 \longrightarrow^* M$ with $|Car(M)| \geq 2$ and $Tok(M) > c$. (Any pumping cycle for p in 3. contains $p' \neq p$.) Hence we cannot have $\alpha_0 \sim M_0$ due to Proposition 12.

($\Leftarrow$) Suppose we have an nBPP process (M_0, Δ) where the conditions 1.,2.,3. are satisfied. We show how an appropriate (α_0, Σ) can be constructed. Since all three conditions hold, the only unbounded places in (M_0, Δ) are growing SF-places. Moreover, if some growing SF-place p is reachable from M_0 then $Tok(M_0) = 1$ and each transition sequence reaching p just moves the token into p without creating new tokens on the way.

We can construct the usual reachability graph for M_0, with the exception that the "all-in-one-SF" markings p^k are taken as "frozen" – we construct no successors for them. The thus arising *basic LTS* is necessarily finite, and we can view its states as BPA-variables; each non-frozen marking M is viewed as a variable A_M, with the obvious rewriting rules.

To finish the construction, we introduce a variable I_p for each SF-place p together with appropriate rewriting rules.

More formally, for (M_0, Δ) we could construct nBPA system $\Sigma = (\mathcal{F} \cup \mathcal{I}, \mathcal{A}, \Gamma)$ where $\mathcal{F} = \{A_M \mid M \in \mathcal{M}_{nf}\}$ (where $\mathcal{M}_{nf} = \{M_1, M_2, \ldots, M_m\}$ is the set of non-frozen markings reachable from M_0), $\mathcal{I} = \{I_p \mid p \in P_{SF}\}$ (where $P_{SF} = \{p_1, p_2, \ldots, p_\ell\}$ is the set of SF-places of Δ), and Γ contains the corresponding rewriting rules.

Note that each rule in Γ is of one of the following three forms: $A_M \stackrel{a}{\longrightarrow} A_{M'}$, $A_M \stackrel{a}{\longrightarrow} (I_p)^k$, or $I_p \stackrel{a}{\longrightarrow} (I_p)^k$, where $A_M, A_{M'} \in \mathcal{F}$, $I_p \in \mathcal{I}$, and $k \in \mathbb{N}$ (this includes also rules of the form $A_M \stackrel{a}{\longrightarrow} \varepsilon$ and $I_p \stackrel{a}{\longrightarrow} \varepsilon$). Configuration α_0 corresponding to M_0 will be A_{M_0} (or $(I_{p_0})^k$ when all k tokens in M_0 are in one SF-place p_0). Note that each configuration α reachable from α_0 is either of the form A_M or $(I_p)^k$, and we have $(\alpha_0, \Sigma) \sim (M_0, \Delta)$. □

We note that the conditions in Lemma 20 can be checked by straightforward standard algorithms, linear in the size of Δ in prime form (which means $O(n^3)$ if Δ is not in prime form). We thus have the following corollary.

Corollary 21. *The problem to decide if a given normed BPP process (not necessarily in prime form) is bisimilar to some (unspecified) normed BPA process can be solved in time $O(n^3)$.*

5.2. Polynomial algorithm for nBPA-nBPP-BISIM

Assume an instance of nBPA-nBPP-BISIM, i.e., nBPA process (α_0, Σ) and nBPP process (M_0, Δ). The polynomial algorithm for nBPA-nBPP-BISIM works as follows.

It first transforms (M_0, Δ) to bisimilar (M_0', Δ') where Δ' is in prime form; recall Theorem 9. Note that nothing special is assumed about (α_0, Σ) and it is not transformed to any special form. The algorithm then starts to build nBPA Σ' for (M_0', Δ') as described in the proof of Lemma 20 by building the set $\mathcal{M}_{nf}$ of non-frozen states. If it discovers that the number of elements of

$\mathcal{M}_{nf}$ exceeds $4y^2$, where y is the maximum of $\{|V_\Sigma|, |P_{\Delta'}|\}$, then the algorithm stops with the answer $\alpha_0 \not\sim M_0$; this is correct due to Theorem 10. Note that it is not necessary to test the conditions of Lemma 20 explicitly in the algorithm because if any of these conditions is violated, the number of non-frozen markings is infinite, which means that the number of constructed elements of $\mathcal{M}_{nf}$ necessarily exceeds $4y^2$ and the algorithm stops with the correct answer.

Remark. Generally the size of Δ' is $O(n^3)$ in the size n of the nBPA-nBPP-BISIM-instance. But since $|P_{\Delta'}| \leq |P_\Delta|$ (recall Theorem 9), the bound $4y^2$ is in $O(n^2)$.

If the number of elements of $\mathcal{M}_{nf}$ does not exceed $4y^2$, the algorithm finishes the construction of Σ'. However, it does not construct Σ' explicitly but rather a succinct representation of it where the right hand sides of rules of the form $(I_p)^k$ are represented as pairs (I_p, k) where k is written in binary (note that $O(n)$ bits are sufficient for k).

Our aim is to apply the polynomial time algorithm from [8] or [9] to decide if $\alpha_0 \sim \alpha'_0$. However, there is a small technical difficulty since this algorithm expects "usual" nBPA, not nBPA in the succinct form described above. This can be handled by adding special variables I_p^1, I_p^2, I_p^4, I_p^8, ... $I_p^{2^m}$ for each $I_p \in \mathcal{I}$ and sufficiently large m (in $O(n)$); the rules are adjusted in a straightforward way (note that there will be at most $O(m)$ variables on the right hand side of each rewriting rule after this transformation).

The size of the constructed nBPA is clearly polynomial with respect to the size of the original instance of the problem and the algorithm from [8] or [9] can be applied.

So we obtained our main theorem:

Theorem 22. *There is a polynomial-time algorithm deciding whether* $(\alpha_0, \Sigma) \sim (M_0, \Delta)$ *where* Σ *is a normed BPA and* Δ *a normed BPP.*

Since (α'_0, Σ') is in a very special form (it is a finite state system (FS) extended with "SF-tails"), it is in fact not necessary to use the above mentioned general algorithm. Instead we can use a specialized and more efficient algorithm described in the next section.

6. An algorithm deciding nBPA-nBPP-BISIM in $O(n^7)$

The aim of this section is to provide a self-contained algorithm for nBPA-nBPP-BISIM. It is inspired by the ideas used, e.g., in the proofs in [14, 16, 9]; being tailored to our specific setting, the algorithm allows to derive the upper bound $O(n^7)$. In Subsection 6.1 we fix some notation and in Subsection 6.2 we deal with the simple subcase of the "single final" configurations. Subsection 6.3 can be seen as an adaptation of the bisimulation base construction from, e.g., [14, 16]. Subsection 6.4 recalls a useful fact on boolean equation systems, which was also used in [9]; the respective application to our case is described in Subsection 6.5. Subsection 6.6 then presents the overall algorithm. We can note that the described algorithm does not use the fact that nBPA processes have unique decomposition property.

6.1. Notation

Assume we have an nBPA process (α_0, Σ) and an nBPP process (M_0, Δ) (not necessarily in prime form) from the instance of nBPA-nBPP-BISIM, and the nBPA process (α'_0, Σ') obtained from (M_0, Δ) as described in the previous section (with $V_{\Sigma'} = \mathcal{F} \cup \mathcal{I}$) stored using the succinct representation described above (the right hand sides of the form $(I_p)^i$ are stored as pairs (I_p, i) with i represented in binary).

In the rest of the section, we assume the following:

- n is the size (in bits) of the original instance of nBPA-nBPP-BISIM,
- m is the size of Σ (note that $m < n$, $|V_\Sigma| < m$, and m is greater than the sum of lengths of the right hand sides of the rules of Σ),
- $k = |V_{\Sigma'}| = |\mathcal{F}| + |\mathcal{I}|$,
- ℓ is the total number of the rules of Σ'.

It is clear from the previous discussion that $|\mathcal{F}| \in O(n^2)$, $|\mathcal{I}| < n$, and $k \in O(n^2)$. Since each reachable configuration α of (α'_0, Σ') is bisimilar to some marking of Δ, the number of transitions enabled in α is bounded by the number of transitions of Δ, and so it is less than n. This means that $\ell \in O(n^3)$.

Recall that all reachable configurations of (α'_0, Σ') are either of the form A_M or $(I_p)^i$ ($A_M \in \mathcal{F}$, $I_p \in \mathcal{I}$). We denote the set of all such configurations by $Conf(\Sigma')$, i.e.,

$$Conf(\Sigma') = \mathcal{F} \cup \{(I_p)^i \mid I_p \in \mathcal{I},\, i \geq 0\}\,.$$

Without loss of generality we assume $\mathcal{I} \neq \emptyset$, which ensures that $\varepsilon \in Conf(\Sigma')$.

Let $V_{all} = V_\Sigma \cup V_{\Sigma'}$. We easily note that the values $\|X\|$, $\|\alpha\|$ for each $X \in V_{all}$, and each α such that $X \longrightarrow \alpha$ can be written in $O(n)$ bits.

6.2. Characterization of configurations bisimilar to $(I_p)^i$

The following proposition allows us to characterize the set of configurations from V^*_{all} bisimilar to $(I_p)^i$ where $I_p \in \mathcal{I}$ and $i \geq 0$.

Proposition 23. *For each $I_p \in \mathcal{I}$ there is a set $Class(I_p) \subseteq V_{all}$ such that for each $\alpha \in V^*_{all}$ we have $\alpha \sim (I_p)^i$ iff $\alpha \in Class(I_p)^*$ and $\|\alpha\| = i$.*

Proof. We construct a set $Class(I_p)$ as the maximal subset of V_{all} such that each $X \in Class(I_p)$ can perform exactly the same actions with the same changes on norm as I_p, and can be rewritten only to variables from $Class(I_p)$ (i.e., $X \xrightarrow{a} \beta$ implies $\beta \in (Class(I_p))^*$, and $I_p \xrightarrow{a} (I_p)^i$ iff $X \xrightarrow{a} \beta$ for some $\beta \in (Class(I_p))^*$ such that $\|\beta\| - \|X\| = i - 1$). □

Note that the classes $Class(I_p)$ for $I_p \in \mathcal{I}$ can be easily computed in polynomial time and can be precomputed at the beginning. This gives us a fast (polynomial) test for checking if $\alpha \sim (I_p)^i$.

6.3. Bisimulation base

We start with some observations leading to the technical notions defined below. Suppose we want to check if $\alpha \sim A_M$ for some $\alpha \in V^*_{all}$ and $A_M \in \mathcal{F}$ where $\alpha = X\alpha'$ for some $X \in V_{all}$. If $X\alpha' \sim A_M$ then any norm reducing sequence $X\alpha' \longrightarrow^*_R \alpha'$ must be matched by some norm reducing sequence $A_M \longrightarrow^*_R \beta$ such that $\alpha' \sim \beta$. Obviously, β is either of the form $A_{M'}$ (for some $A_{M'} \in \mathcal{F}$) or $(I_p)^i$ (for some $I_p \in \mathcal{I}$). Since $\alpha' \sim \beta$ and $\sim$ is a congruence, we have $X\beta \sim A_M$. On the other hand, from $X\beta \sim A_M$ and $\alpha' \sim \beta$ follows

$X\alpha' \sim A_M$. So we see that $X\alpha' \sim A_M$ iff there is some $\beta \in \mathit{Conf}(\Sigma')$ such that $X\beta \sim A_M$ and $\alpha' \sim \beta$.

This allows us to construct a bisimulation base, i.e. a succinct representation of $\sim$ on pairs of (reachable) configurations of (α_0, Σ) and (α_0', Σ'). The base is a finite set (of polynomial size) containing some bisimilar pairs from which all other bisimilar pairs can be generated.

We start by defining (an overapproximation)

$$\begin{array}{rcl} \mathcal{B}_0 & = & \{(X\alpha, A) \mid X \in V_\Sigma,\ \alpha \in \mathit{Conf}(\Sigma'),\ A \in \mathcal{F},\ \|X\alpha\| = \|A\|\} \\ & \cup & \{(\alpha, A) \mid \alpha \in \mathit{Conf}(\Sigma'),\ A \in \mathcal{F},\ \|\alpha\| = \|A\|\}\,. \end{array}$$

Note that $\mathcal{B}_0$ is finite since i in $(XI^i, A) \in \mathcal{B}_0$ is determined by X, I, A and the requirement $\|XI^i\| = \|A\|$ and i can be computed as $i = \|A\| - \|X\|$ (and similarly i in $(I^i, A) \in \mathcal{B}_0$). So $\mathcal{B}_0$ contains at most $(|V_\Sigma|+1)\cdot(|\mathcal{F}|+|\mathcal{I}|)\cdot|\mathcal{F}| = O(mk^2)$ elements.

For each $\mathcal{B} \subseteq \mathcal{B}_0$ we define the set $\mathit{Closure}(\mathcal{B})$ as the least subset of $\{(\gamma\alpha, \alpha') \mid \gamma \in V_\Sigma,\ \alpha, \alpha' \in \mathit{Conf}(\Sigma')\}$ satisfying the following properties:

(1) $\mathcal{B} \subseteq \mathit{Closure}(\mathcal{B})$.

(2) Let $X \in V_\Sigma$, $\gamma \in V_\Sigma^+$, $\alpha \in \mathit{Conf}(\Sigma')$, and $A \in \mathcal{F}$. Then $(X\gamma\alpha, A) \in \mathit{Closure}(\mathcal{B})$ iff $\exists \alpha' \in \mathit{Conf}(\Sigma') : (X\alpha', A) \in \mathcal{B} \ \wedge\ (\gamma\alpha, \alpha') \in \mathit{Closure}(\mathcal{B})$.

(3) Let $\gamma \in V_\Sigma^*$, $\alpha \in \mathit{Conf}(\Sigma')$, $I \in \mathcal{I}$, and $i \geq 0$. Then $(\gamma\alpha, I^i) \in \mathit{Closure}(\mathcal{B})$ iff $\gamma\alpha \sim I^i$.

The aim of the algorithm is to find the *bisimulation base*

$$\mathcal{B}_\sim = \{(\alpha, A) \mid (\alpha, A) \in \mathcal{B}_0,\ \alpha \sim A\}$$

which can be used as a finite representation of bisimilar pairs in the sense of the following proposition.

Proposition 24. *$\mathit{Closure}(\mathcal{B}_\sim)$ coincides with the set $\{(\gamma\alpha, \beta) \mid \gamma \in V_\Sigma^*,\ \alpha, \beta \in \mathit{Conf}(\Sigma'),\ \gamma\alpha \sim \beta\}$.*

Proof idea. Follows directly from the definition of $\mathit{Closure}(\mathcal{B}_\sim)$ using induction on $|\gamma|$. □

Remark. Note that for each $\gamma \in V_\Sigma^*$ and $\beta \in \mathit{Conf}(\Sigma')$ we have $(\gamma, \beta) \in \mathit{Closure}(\mathcal{B}_\sim)$ iff $\gamma \sim \beta$.

Given a set $\mathcal{B} \subseteq \mathcal{B}_0$ and a pair $(\alpha, \alpha') \in \mathcal{B}$, we say (α, α') *satisfies expansion in* $\mathcal{B}$ if the two following conditions are satisfied for each $a \in \mathcal{A}$:

- $\forall \beta : \alpha \xrightarrow{a} \beta \Rightarrow (\exists \beta' : \alpha' \xrightarrow{a} \beta' \wedge (\alpha', \beta') \in \mathit{Closure}(\mathcal{B}))$, and
- $\forall \beta' : \alpha' \xrightarrow{a} \beta' \Rightarrow (\exists \beta : \alpha \xrightarrow{a} \beta \wedge (\alpha', \beta') \in \mathit{Closure}(\mathcal{B}))$.

By $\mathcal{E}(\mathcal{B})$ we denote the set of those pairs in $\mathcal{B}$ that satisfy expansion in $\mathcal{B}$. Notice that the mapping $\mathcal{E}$ is monotonic, i.e., $\mathcal{B} \subseteq \mathcal{B}'$ implies $\mathcal{E}(\mathcal{B}) \subseteq \mathcal{E}(\mathcal{B}')$. Note also that $\mathcal{B}_\sim = \mathcal{E}(\mathcal{B}_\sim)$. Consider now the sequence

$$\mathcal{B}_0 \supseteq \mathcal{B}_1 \supseteq \mathcal{B}_2 \supseteq \cdots$$

where $\mathcal{B}_{i+1} = \mathcal{E}(\mathcal{B}_i)$ for $i \geq 0$. Since $\mathcal{B}_\sim \subseteq \mathcal{B}_0$ and due to monotonicity of $\mathcal{E}$ we obtain $\mathcal{B}_\sim \subseteq \mathcal{B}_i$ for each $i \geq 0$.

Since $\mathcal{B}_0$ is finite, there must be a fixpoint $\mathcal{B}_i = \mathcal{E}(\mathcal{B}_i)$ for some $i \geq 0$. As follows from the following proposition (which can be easily checked), this fixpoint coincides with $\mathcal{B}_\sim$:

Proposition 25. *If $\mathcal{B} = \mathcal{E}(\mathcal{B})$ then $\mathit{Closure}(\mathcal{B})$ is a bisimulation.*

In fact, it is not necessary to compute the sequence $\mathcal{B}_0, \mathcal{B}_1, \mathcal{B}_2, \ldots$ as it was done in [14, 16]. Instead, we can use the idea from [9] of a reduction to the problem of finding a (unique) maximal solution of a certain set of boolean equations, which was used there in the algorithm for deciding bisimilarity on normed BPA. The idea considerably simplifies the complexity analysis and gives a better complexity bound than would be obtained by a straightforward analysis of the algorithm based on the computation of the fixpoint.

6.4. Boolean equation systems

Let $\mathcal{V} = \{x_1, x_2, \ldots, x_r\}$ be a (finite) set of boolean variables. A *boolean equation system* is a set of equations of the form

$$x_i = \varphi_i(x_1, x_2, \ldots, x_r)$$

where each φ_i is a monotonic boolean formula over $\mathcal{V}$, i.e., a boolean formula constructed using variables from $\mathcal{V}$, and symbols $\wedge$, $\vee$, $\top$, and $\bot$ (symbols $\top$

and $\bot$ denote the formulas that are always true or always false, respectively). In particular, the negation $\neg$ can not be used in φ_i. A valuation ν is a mapping $\nu : \mathcal{V} \to \{\texttt{true}, \texttt{false}\}$; it can be extended to formulas in the obvious manner. A valuation ν is a solution of a given boolean equation system if $\nu(x_i) = \nu(\varphi_i)$ for each i.

On valuations we can define the partial order $\sqsubseteq$ such that $\nu \sqsubseteq \nu'$ iff $\nu(x) = \texttt{true}$ implies $\nu'(x) = \texttt{true}$ (for each $x \in \mathcal{V}$). A valuation ν is the maximal solution of a boolean equation system if it is the solution of the equation system and it is maximal wrt $\sqsubseteq$. It follows from the well-known Knaster-Tarski fixpoint theorem [17] that every boolean equation system has a unique maximal solution.

The following simple fact, also used in [9], is crucial for obtaining an efficient algorithm for the computation of $\mathcal{B}_\sim$:

Proposition 26. *Given a boolean equation system, its maximal solution can be found in time linear wrt the size of the system.*

Proof idea. One possibility, how to get a linear time algorithm for finding the maximal solution of a boolean equation system, is to construct a boolean circuit whose inputs correspond to variables in $\mathcal{V}$ and outputs to values of φ_i for each i, to assign `true` to all gates except those that correspond to $\bot$, and then propagate values `false` through the circuit. In particular, when the output corresponding to some φ_i is set to `false`, the input gate corresponding to x_i is set to `false`. □

6.5. Construction of the boolean equation system for finding $\mathcal{B}_\sim$

We describe how to construct a boolean equation system *BES* such that the maximal solution ν_{max} of *BES* represents $\mathcal{B}_\sim$. Variables of *BES* correspond to pairs (α, β) of configurations; the variable corresponding to (α, β) is denoted $x_{(\alpha,\beta)}$. The system *BES* is constructed so that for each variable $x_{(\alpha,\beta)}$ of *BES*, $\nu_{max}(x_{(\alpha,\beta)}) = \texttt{true}$ iff $\alpha \sim \beta$.

There are variables of two types in *BES*:

Type 1: For each $(\alpha, \beta) \in \mathcal{B}_0$ there is a boolean variable $x_{(\alpha,\beta)}$.

Type 2: For each $\gamma \in V_\Sigma^+$, $\alpha \in \mathit{Conf}(\Sigma')$ and $A \in \mathcal{F}$ such that $\|\gamma\alpha\| = \|A\|$ and γ is a suffix of the right hand side of some rule of Σ (i.e.,

$(X \xrightarrow{a} \delta\gamma) \in \Gamma_\Sigma$ for some X, a and δ) such that $|\gamma| > 1$, there is a boolean variable $x_{(\gamma\alpha,A)}$

Note that there are $|\mathcal{B}_0| = O(mk^2)$ variables of type 1, and since the number of suffixes of the right-hand sides of rules of Σ' is less than m, there can be at most mk^2 variables of type 2.

Before defining formulas for all variables in *BES*, we define auxiliary formulas $g(\alpha, \beta)$ for each α, β where $\|\alpha\| = \|\beta\|$ (formulas $g(\alpha, \beta)$ are used as subformulas in formulas in *BES*):

- If β is of the form I^i for some $I \in \mathcal{I}$: if $\alpha \sim \beta$ then $g(\alpha, \beta) = \top$ else $g(\alpha, \beta) = \bot$. (Recall $\alpha \sim I^i$ iff $\alpha \in \mathit{Class}(I)^*$ and $\|\alpha\| = i$.)
- If $\beta \in \mathcal{F}$ then $g(\alpha, \beta) = x_{(\alpha,\beta)}$. (Assuming that the variable $x_{(\alpha,\beta)}$ exists in *BES*, which will be ensured in the following constructions.)

The system *BES* contains the following equation for each variable $x_{(\alpha,\beta)}$ of type 1:

$$x_{(\alpha,\beta)} = \bigwedge_{\alpha \xrightarrow{a} \alpha'} \left(\bigvee_{\substack{\beta \xrightarrow{b} \beta' \\ \text{where } a=b \\ \text{and } \|\alpha'\|=\|\beta'\|}} g(\alpha', \beta') \right) \wedge \bigwedge_{\beta \xrightarrow{b} \beta'} \left(\bigvee_{\substack{\alpha \xrightarrow{a} \alpha' \\ \text{where } a=b \\ \text{and } \|\alpha'\|=\|\beta'\|}} g(\alpha', \beta') \right)$$

The equation expresses that every transition $\alpha \xrightarrow{a} \alpha'$ enabled in α must be matched by some transition $\beta \xrightarrow{a} \beta'$ enabled in β and vice versa, recall the definition of $\mathcal{E}$. (Note that all subformulas $g(\alpha, \beta)$ are defined correctly in the above formula.)

For each variable $x_{(X\alpha,A)}$ of type 2 (where necessarily $X \in V_\Sigma$ and α starts with a symbol from V_Σ), the system *BES* contains the equation

$$x_{(X\alpha,A)} = \bigvee_{\substack{B \in \mathcal{F} \\ \text{s.t. } \|B\|=\|\alpha\|}} (g(XB, A) \wedge g(\alpha, B)) \vee \bigvee_{I \in \mathcal{I}} \left(g(XI^{\|\alpha\|}, A) \wedge g(\alpha, I^{\|\alpha\|}) \right) .$$

This formula directly corresponds to point (2) of the definition of $\mathit{Closure}(\mathcal{B})$.

To estimate the sizes of the formulas in *BES*, it is obviously sufficient to estimate the number of occurrences of subformulas $g(\alpha, \beta)$ in these formulas. (Note that the size of each $g(\alpha, \beta)$ is $O(1)$.)

Let us consider formulas for variables of type 1 of the form $x_{(X\alpha,A)}$ where $X \in \mathcal{V}_\Sigma$. The rules that can be used for possible transitions in $X\alpha$ depend only on X. If we count the total number of pairs of rules $X \xrightarrow{a} \gamma$ and $A \xrightarrow{a} \beta$ for all $X \in V_\Sigma$, $A \in \mathcal{F}$, we can see that there is at most $m\ell$ such pairs of rules. Each such pair is used in at most k formulas (there are at most k possible values of α), and it is used at most twice in each formula. So the total size of formulas for the variables of type 1 of the above mentioned form is at most $O(m\ell k)$. Similarly, the total size of formulas for the variables of type 1 of the form $x_{(\alpha,A)}$ where $\alpha \in \mathit{Conf}(\Sigma')$ is at most $O(\ell^2)$.

It is clear that the size of each formula for a variable of type 2 is $O(|\mathcal{F}|+|\mathcal{I}|) = O(k)$. Since there are at most mk^2 variables of type 2, the total size of their formulas is $O(mk^3)$.

Summing the sizes of the formulas in *BES* we obtain:

Proposition 27. *The size of BES is* $O(mk^3 + m\ell k + \ell^2) = O(n^7)$.

6.6. The overall algorithm

Theorem 28. *There is an algorithm solving* nBPA-nBPP-BISIM *in time* $O(n^7)$.

Proof. The algorithm works as described above. It transforms the given nBPP (M_0, Δ) into prime form and generates (a succinct representation of) nBPA (α'_0, Σ') from it. If the construction of (α'_0, Σ') is finished (i.e., the algorithm does not stop with the negative answer), the corresponding boolean equation system *BES* of size $O(n^7)$ (recall Proposition 27) is constructed and the algorithm finds its maximal solution ν_{max} in time $O(n^7)$ (recall Proposition 26). The algorithm then checks if $\nu_{max}(x_{(\alpha_0,\alpha'_0)}) = \texttt{true}$ (without loss of generality we can assume that $\alpha_0 \in V_\Sigma$, $\alpha'_0 \in \mathcal{F}$ and so *BES* contains the variable $x_{(\alpha_0,\alpha'_0)}$) which gives the answer for the original instance of nBPA-nBPP-BISIM.

Before the construction of *BES*, the rules of Σ and Σ' can be partitioned according to their labels and the changes on norms they cause. Norms for all $X \in V_{all}$ and for all suffixes of right hand sides of rules of Σ can be

precomputed. Note that there are at most $O(n^5)$ different subformulas of the form $g(\alpha, \beta)$ that occur in formulas for variables of type 2 and that for every such pair the subformula $g(\alpha, \beta)$ can be precomputed in time $O(n^2)$. Using all this precomputed information, the system *BES* can be constructed in time $O(n^7)$.

All other steps of the algorithm (the transformation to prime form, the generation of Σ', the precomputation of the sets $Class(I)$ for all $I \in \mathcal{I}$ and the precomputation of all other necessary information described above) can be obviously done in time $O(n^7)$. □

After ν_{max} has been computed, it can be used for deciding efficiently if $\gamma \sim A$ for all $\gamma \in V_\Sigma$, $A \in \mathcal{F}$. Just note that for each suffix γ' of γ we can find all $\beta \in Conf(\Sigma')$ such that $\gamma' \sim \beta$ (in fact there is always at most one such β due to the fact that all configurations in $Conf(\Sigma')$ are pairwise non-bisimilar) assuming this information was already computed for all its proper suffixes.

Remark. The above algorithm can be used for deciding bisimilarity between a given nBPA (of size m) and a finite state system (with k states and ℓ transitions) and the running time of the algorithm is $O(mk^3 + m\ell k + \ell^2) = O(n^4)$ in this case (where n is the size of the whole instance). In fact, the algorithm can be easily adapted for the case when the BPA and the FS in the instance are not required to be normed (as in [14, 16]) without affecting its complexity. The more general problem of deciding weak bisimilarity on a given BPA and FS process was considered in [14] and the algorithm presented there has running time $O(m^5(k+\ell)^7) = O(n^{12})$. The special case of the strong bisimilarity was not analyzed there and we are not aware of any tighter results concerning its complexity.

7. Conclusions

By a detailed analysis and a combination of several simple ideas and observations we have managed to lower the exponential time complexity upper bound to polynomial when deciding bisimilarity between normed BPA and BPP processes. We think that a similar closer look should also allow to give a more precise complexity bound for the general case of (unnormed) BPA and BPP processes. This in turn might help to build a better understanding of the so far open problem for the class PA.

References

[1] 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, Ch. 9, pp. 545–623.

[2] 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://www.brics.dk/∼srba/roadmap/).

[3] Y. Hirshfeld, M. Jerrum, Bisimulation equivalence is decidable for normed process algebra, in: Proceedings of 26th International Colloquium on Automata, Languages and Programming (ICALP'99), Vol. 1644 of Lecture Notes in Computer Science, Springer-Verlag, 1999, pp. 412–421.

[4] O. Burkart, D. Caucal, B. Steffen, An elementary decision procedure for arbitrary context-free processes, in: Proceedings of the 20th International Symposium on Mathematical Foundations of Computer Science (MFCS'95), Vol. 969 of Lecture Notes in Computer Science, Springer-Verlag, 1995, pp. 423–433.

[5] J. Srba, Strong bisimilarity and regularity of Basic Process Algebra is PSPACE-hard, in: Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP'02), Vol. 2380 of Lecture Notes in Computer Science, Springer, 2002, pp. 716–727.

[6] P. Jančar, Strong bisimilarity on Basic Parallel Processes is PSPACE-complete, in: Proceedings of 18th IEEE Symposium on Logic in Computer Science (LICS 2003), IEEE Computer Society, 2003, pp. 218–227.

[7] J. Srba, Strong bisimilarity and regularity of Basic Parallel Processes is PSPACE-hard, in: Proceedings of 19th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2002), Vol. 2285 of Lecture Notes in Computer Science, Springer, 2002, pp. 535–546.

[8] Y. Hirshfeld, M. Jerrum, F. Moller, A polynomial algorithm for deciding bisimilarity of normed context-free processes, Theoretical Comput. Sci. 158 (1996) 143–159.

[9] S. Lasota, W. Rytter, Faster algorithm for bisimulation equivalence of normed context-free processes, in: Proceedings of 31st International Symposium on Mathematical Foundations of Computer Science (MFCS 2006), Vol. 4162 of Lecture Notes in Computer Science, Springer-Verlag, 2006, pp. 646–657.

[10] Y. Hirshfeld, M. Jerrum, F. Moller, A polynomial-time algorithm for deciding bisimulation equivalence of normed Basic Parallel Processes, Mathematical Structures in Computer Science 6 (1996) 251–259.

[11] 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.

[12] I. Černá, M. Křetínský, A. Kučera, Comparing expressibility of normed BPA and normed BPP processes, Acta Informatica 36 (1999) 233–256.

[13] P. Jančar, A. Kučera, F. Moller, Deciding bisimilarity between BPA and BPP processes, in: Proceedings of the 14th International Conference on Concurency Theory (CONCUR 2003), Vol. 2761 of Lecture Notes in Computer Science, Springer-Verlag, 2003, pp. 159–173.

[14] A. Kučera, R. Mayr, Weak bisimilarity between finite-state systems and BPA or normed BPP is decidable in polynomial time, Theoretical Computer Science 270 (2002) 667–700.

[15] P. Jančar, M. Kot, Z. Sawa, Normed BPA vs. normed BPP revisited, in: Proceedings of the 19th International Conference on Concurency Theory (CONCUR 2008), Vol. 5201 of Lecture Notes in Computer Science, Springer, 2008, pp. 434–446.

[16] A. Kučera, R. Mayr, A generic framework for checking semantic equivalences between pushdown automata and finite-state automata, in: IFIP TCS, Kluwer, 2004, pp. 395–408.

[17] A. Tarski, A lattice-theoretical fixpoint theorem and its applications, Pacific Journal of Mathematics 5 (2) (1955) 285–309.