Weak Bisimulation Approximants

Will Harwood and Faron Moller

Department of Computer Science, Swansea University

Abstract. ***Bisimilarity***, a canonical notion of equivalence between processes, is defined co-inductively, but it may be approached – and even reached – by its (transfinite) inductively-defined ***approximants***. For arbitrary processes this approximation may need to climb arbitrarily high through the infinite ordinals before stabilising. In this paper we consider a simple process algebra, the Basic Parallel Processes (BPP), and investigate the level at which the approximation becomes the thing itself.

1 Introduction

There has been great interest of late in the development of techniques for deciding equivalences between infinite-state processes, particularly for the question of bisimilarity between processes generated by some type of term algebra. Several surveys of this developing area have been published, beginning with [13], and there is now a chapter in the Handbook of Process Algebra dedicated to the topic [1], as well as a website devoted to maintaining an up-to-date comprehensive overview of the state-of-the-art.

While questions concerning strong bisimilarity have been successfully addressed, techniques for tackling the question of weak bisimilarity, that is, when unobservable transitions are allowed, are still lacking, and many open problems remain. The main difficulty arising when considering weak bisimilarity is that processes become infinite-branching: at any point in a computation, a single action can result in a transition to any one of an infinite number of next states. Common finiteness properties fail due to this; in particular, bisimilarity can no longer be characterised by its finite approximations in the way that it can for finite-branching processes. For arbitrary infinite-branching processes, we may need to climb arbitrarily high through the transfinite approximations to bisimilarity before reaching the bisimulation relation itself.

In this paper we consider the problem of weak bisimilarity for so-called Basic Parallel Processes (BPP), a simple model of concurrent processes. These correspond to commutative context-free processes, or equivalently to communication-free Petri nets. The question as to the decidability of weak bisimilarity between BPP processes remains unsolved. It has recently been shown that the problem is

at least PSPACE-hard [16], even in the restricted case of so-called normed BPP, but this sheds no light one way or the other as to decidability. Jančar suggests in [11] that the techniques he uses there to establish PSPACE-completeness of strong bisimilarity for BPP might be exploited to give a decision procedure for weak bisimilarity, but this conjecture remains unsubstantiated.

It has long been conjectured that for BPP, weak bisimilarity is characterised by its $(\omega\times 2)$-level approximation. Such a result could provide a way to a decision procedure. However, no nontrivial approximation bound has before now been established. In this paper we provide a non-trivial countable bound on the approximation: for a BPP defined over k variables, weak bisimilarity is reached by the ω^k level; weak bisimilarity is thus reached by the ω^ω level for any BPP.

Our argument is based on a new constructive proof of Dickson's Lemma which provides an ordinal bound on the sequences described by the Lemma. This proof is presented in Section 2 of the paper. After this, the definitions necessary for the remainder of the paper are outlined in Section 3 along with a variety of results, and our results on BPP are presented in Section 4.

2 Ordinal Bounds for Dickson's Lemma

An n-tuple $\vec{\boldsymbol{y}}=(y_1,\ldots,y_n)\in\mathbb{N}^n$ ***dominates*** another n-tuple $\vec{\boldsymbol{x}}=(x_1,\ldots,x_n)\in\mathbb{N}^n$ if $\vec{\boldsymbol{x}}\leq\vec{\boldsymbol{y}}$, where $\leq$ is considered pointwise. A sequence of n-tuples is a ***non-dominating sequence*** over $\mathbb{N}^n$ if no element of the sequence dominates any of its predecessors in the sequence. A rooted tree with nodes labelled by n-tuples from $\mathbb{N}^n$ is a ***non-dominating tree*** over $\mathbb{N}^n$ if the sequence of labels along any path through the tree is a non-dominating sequence.

Dickson's Lemma [5] asserts that there can be no infinite non-dominating sequences.

Lemma 1 (Dickson's Lemma). *All non-dominating sequences are finite. That is, given an infinite sequence of vectors $\vec{\boldsymbol{x}}_1,\vec{\boldsymbol{x}}_2,\vec{\boldsymbol{x}}_3,\ldots\in\mathbb{N}^n$ we can always find indices i,j with $i<j$ such that $\vec{\boldsymbol{x}}_i\leq\vec{\boldsymbol{x}}_j$.*

The standard proof of this lemma uses a straightforward induction on n: for the base case, any sequence of decreasing natural numbers must be finite; and for the induction step, from an infinite sequence of n-tuples you extract an infinite subsequence in which the last components are nondecreasing (either constant or increasing), and then apply induction on the sequence of $(n-1)$-tuples arising when ignoring these last components.

The problem with this proof is that it is nonconstructive; in particular, it gives no clue as to the ordinal bound on the lengths of non-dominating sequences.

The difficulty with determining an ordinal bound comes from the fact that the domination order is not a total order on n-tuples (as opposed, for example, to lexicographical order). We provide here an alternative constructive proof from which we can extract an ordinal bound on the lengths of such sequences.

Lemma 2 (Constructive Dickson's Lemma). *The ordinal bound on non-dominating sequences of n-tuples is ω^n.*

Proof. That ω^n is a lower bound is clear: we can construct non-dominating sequences of n-tuples from $\mathbb{N}^n$ which decrease arbitrarily slowly with respect to the lexicographical ordering $<_{\text{lex}}$ on $\mathbb{N}^n$. That it is an upper bound will follow from the construction of a function $f_n : (\mathbb{N}^n)^+ \to \mathbb{N}^n$ on non-empty sequences of n-tuples which satisfies the following property: (here, $*$ represents the operation of concatenating sequences)

> If $\vec{\boldsymbol{u}} * \langle \vec{\boldsymbol{x}} \rangle$ is a non-dominating sequence of n-tuples, and $\vec{\boldsymbol{u}}$ is itself non-empty, then $f_n(\vec{\boldsymbol{u}} * \langle \vec{\boldsymbol{x}} \rangle) <_{\text{lex}} f_n(\vec{\boldsymbol{u}})$.

We shall inductively define these functions f_n. The base case is straightforward: we can define f_1 by

$$f_1(\langle x_1, \ldots, x_k \rangle) = x_k.$$

A non-dominating sequence of natural numbers is simply a decreasing sequence, which has ordinal bound ω.

For illustrative purposes we carry out the construction of the function f_2 for sequences of pairs, and later generalise our construction to sequences of n-tuples.

Given a non-empty finite sequence $\vec{\boldsymbol{u}} = \langle (x_1, y_1), \ldots, (x_k, y_k) \rangle$ of pairs, define

- $\text{MIN}_x(\vec{\boldsymbol{u}}) \;=\; \min\{\, x_i \;:\; 1 \le i \le k \,\}$
- $\text{MIN}_y(\vec{\boldsymbol{u}}) \;=\; \min\{\, y_i \;:\; 1 \le i \le k \,\}$
- $S_2(\vec{\boldsymbol{u}}) \;=\; \Big\{ (x,y) \;:\; \text{MIN}_x(\vec{\boldsymbol{u}}) \le x,\ \text{MIN}_y(\vec{\boldsymbol{u}}) \le y,\ \textit{and} \\ (x_i, y_i) \not\le (x,y) \textit{ for all } i : 1 \le i \le k \Big\}$

$S_2(\vec{\boldsymbol{u}})$ consists of the pairs with which the sequence $\vec{\boldsymbol{u}}$ can be extended without altering the MIN_x and MIN_y values and yet while maintaining non-domination. Note that $S_2(\vec{\boldsymbol{u}})$ must be finite: if we let i and j be such that $x_i = \text{MIN}_x(\vec{\boldsymbol{u}})$ and $y_j = \text{MIN}_y(\vec{\boldsymbol{u}})$, then in order for $(x,y) \not\ge (x_i, y_i)$ and $(x,y) \not\ge (x_j, y_j)$ we must have $x < x_j$ (since $y \ge y_j$) and $y < y_i$ (since $x \ge x_i$).

Suppose that $\vec{v} = \vec{u}{*}\langle(x,y)\rangle$ is a non-dominating sequence, and that $\vec{u}$ is itself non-empty. Then clearly $\text{MIN}_x(\vec{v}) \leq \text{MIN}_x(\vec{u})$ and $\text{MIN}_y(\vec{v}) \leq \text{MIN}_y(\vec{u})$; and if equality holds in both cases then $S_2(\vec{v}) \subsetneq S_2(\vec{u})$ since $S_2(\vec{v}) \subseteq S_2(\vec{u})$ yet $(x,y) \in S_2(\vec{u}) \setminus S_2(\vec{v})$. Thus $|S_2(\vec{v})| < |S_2(\vec{u})|$.

We can then define the function f_2 on non-empty sequences $\vec{u}$ of pairs as follows:

$$f_2(\vec{u}) \;=\; \big\langle\, \text{MIN}_x(\vec{u}){+}\text{MIN}_y(\vec{u}),\, |S_2(\vec{u})| \,\big\rangle$$

If $\vec{u}{*}\langle(x,y)\rangle$ is a non-dominating sequence and $\vec{u}$ is itself non-empty, then by the above argument we must have that $f_2(\vec{u}{*}\langle(x,y)\rangle) <_{\text{lex}} f_2(\vec{u})$.

For the inductive construction of f_n we assume we have constructed the function f_{n-1} as required. For $1 \leq i \leq n$ we define the function

$$\pi_{\text{-}i}(\langle x_1,\ldots,x_n\rangle) \stackrel{\text{def}}{=} \langle x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n\rangle$$

which simply deletes the ith component from the n-tuple $\langle x_1,\ldots,x_n\rangle$. Next, given a non-empty finite sequence $\vec{u} = \langle \vec{x}_1,\ldots,\vec{x}_k\rangle$ of n-tuples, we define the set

$$\text{ND}_{\text{-}i}(\vec{u}) \stackrel{\text{def}}{=} \Big\{\, \langle \pi_{\text{-}i}(\vec{x}_{i_1}),\ldots,\pi_{\text{-}i}(\vec{x}_{i_p})\rangle \;:\; p>0,\; 0<i_1<\cdots<i_p\leq n,\; \textit{and} \\ \langle \pi_{\text{-}i}(\vec{x}_{i_1}),\ldots,\pi_{\text{-}i}(\vec{x}_{i_p})\rangle \text{ is non-dominating} \,\Big\}$$

which consists of the non-dominating subsequences of $(n{-}1)$-tuples of $\vec{u}$ in which the ith components of the n-tuples have been deleted. Finally we make the following definitions:

- $\text{MIN}_{\text{-}i}(\vec{u}) \stackrel{\text{def}}{=} \min_{<_{\text{lex}}}\{\, f_{n-1}(\vec{x}) \;:\; \vec{x} \in \text{ND}_{\text{-}i}(\vec{u}) \,\}$
- $S_n(\vec{u}) \;=\; \Big\{\, \vec{x} \;:\; \text{MIN}_{\text{-}i}(\vec{u}) = \text{MIN}_{\text{-}i}(\vec{u}{*}\langle\vec{x}\rangle) \textit{ for all } i : 1\leq i\leq n,\; \textit{and} \\ \vec{x}_i \not\leq \vec{x} \textit{ for all } i : 1\leq i\leq k \,\Big\}$

$S_n(\vec{u})$ consists of the n-tuples with which the sequence $\vec{u}$ can be extended without altering the $\text{MIN}_{\text{-}i}$ values and yet while maintaining non-domination. Note that $S_n(\vec{u})$ must be finite. To see this, let $i_1,\ldots,i_p$ be such that

$$\text{MIN}_{\text{-}i}(\vec{u}) = f_{n-1}(\langle \pi_{\text{-}i}(\vec{x}_{i_1}),\ldots,\pi_{\text{-}i}(\vec{x}_{i_p})\rangle),$$

and suppose that $\vec{x} \in S_n(\vec{u})$. If $\langle \pi_{\text{-}i}(\vec{x}_{i_1}),\ldots,\pi_{\text{-}i}(\vec{x}_{i_p})\rangle{*}\langle \pi_{\text{-}i}(\vec{x})\rangle$ is non-dominating, then by induction we would get that

$$\begin{aligned}\textsc{min}_{\text{-}i}(\vec{\boldsymbol{u}}{*}\langle\vec{\boldsymbol{x}}\rangle) &\leq_{\text{lex}} f_{n-1}(\langle\pi_{\text{-}i}(\vec{\boldsymbol{x}}_{i_1}),\ldots,\pi_{\text{-}i}(\vec{\boldsymbol{x}}_{i_p})\rangle{*}\langle\pi_{\text{-}i}(\vec{\boldsymbol{x}})\rangle)\\ &<_{\text{lex}} f_{n-1}(\langle\pi_{\text{-}i}(\vec{\boldsymbol{x}}_{i_1}),\ldots,\pi_{\text{-}i}(\vec{\boldsymbol{x}}_{i_p})\rangle) \;=\; \textsc{min}_{\text{-}i}(\vec{\boldsymbol{u}})\end{aligned}$$

contradicting $\vec{\boldsymbol{x}} \in S_n(\vec{\boldsymbol{u}})$. Therefore we must have $\pi_{\text{-}i}(\vec{\boldsymbol{x}}) \geq \pi_{\text{-}i}(\vec{\boldsymbol{x}}_{i_j})$ for some j. But since $\vec{\boldsymbol{x}} \not\geq \vec{\boldsymbol{x}}_{i_j}$ we must then have $(\vec{\boldsymbol{x}})_i < (\vec{\boldsymbol{x}}_{i_j})_i$.

Suppose that $\vec{\boldsymbol{v}} = \vec{\boldsymbol{u}}{*}\langle\vec{\boldsymbol{x}}\rangle$ is a non-dominating sequence, and that $\vec{\boldsymbol{u}}$ is itself non-empty. Then $\textsc{min}_{\text{-}i}(\vec{\boldsymbol{v}}) \leq \textsc{min}_{\text{-}i}(\vec{\boldsymbol{u}})$ for all i (since $\textsc{nd}_{\text{-}i}(\vec{\boldsymbol{u}}) \subseteq \textsc{nd}_{\text{-}i}(\vec{\boldsymbol{v}})$); and if equality holds in all cases then $S_n(\vec{\boldsymbol{v}}) \subsetneq S_n(\vec{\boldsymbol{u}})$ since $S_n(\vec{\boldsymbol{v}}) \subseteq S_n(\vec{\boldsymbol{u}})$ yet $\vec{\boldsymbol{x}} \in S_n(\vec{\boldsymbol{u}}) \setminus S_n(\vec{\boldsymbol{v}})$. Thus $|S_n(\vec{\boldsymbol{v}})| < |S_n(\vec{\boldsymbol{u}})|$.

We can then define the function f_n on non-empty sequences $\vec{\boldsymbol{u}}$ of n-tuples as follows:

$$f_n(\vec{\boldsymbol{u}}) \;=\; \Big(\textstyle\sum_{i=1}^{n} \textsc{min}_{\text{-}i}(\vec{\boldsymbol{u}})\Big){*}\langle|S_n(\vec{\boldsymbol{u}})|\rangle$$

where the sum is taken component-wise. If $\vec{\boldsymbol{u}}{*}\langle\vec{\boldsymbol{x}}\rangle$ is a non-dominating sequence and $\vec{\boldsymbol{u}}$ is itself non-empty, then by the above argument we must have that $f_n(\vec{\boldsymbol{u}}{*}\langle\vec{\boldsymbol{x}}\rangle) <_{\text{lex}} f_n(\vec{\boldsymbol{u}})$. □

2.1 Ordinal Bounds on Trees

This theorem easily extends to trees, where we take the following definition of the height of a well-founded tree.

Definition 1. *The **height** of a well-founded tree rooted at t is defined by*

$$h(t) \stackrel{def}{=} \sup\{\, h(s)+1 \;:\; t \longrightarrow s \,\}.$$

Theorem 1. *If t is a non-dominating tree over $\mathbb{N}^n$, then $h(t) \leq \omega^n$.*

Proof. For each node x of the tree, define $\ell(x) \in \mathbb{N}^n$ as $\ell(x) = f_n(\pi_x)$, where f_n is as defined in the proof of Lemma 2, and π_x is the non-dominating sequence of labels on the path from the root of t to x. It will suffice then to prove that $h(x) \leq \ell(x)$ (viewing $\ell(x)$ as an ordinal) for all nodes x of the tree. This is accomplished by a straightforward induction on $h(x)$:

$$\begin{aligned} h(x) &= \sup\{\, h(y)+1 \;:\; x \to y \,\}\\ &\leq \sup\{\, \ell(y)+1 \;:\; x \to y \,\} \quad \text{(by induction)}\\ &\leq \ell(x). \end{aligned}$$

□

3 Processes and Bisimilarity

A ***process*** is represented by (a state in) a ***labelled transition system*** defined as follows.

Definition 2. *A **labelled transition system (LTS)** is a triple $\mathcal{S} = (S, Act, \rightarrow)$ where S is a set of **states**, Act is a finite set of **actions**, and $\rightarrow \subseteq S \times Act \times S$ is a **transition relation**.*

We write $s \xrightarrow{a} s'$ instead of $(s, a, s') \in \rightarrow$ and we extend this notation to elements of Act^* in the natural way.

It is common to admit silent transitions to model the internal unobservable evolution of a system. In standard automata theory these are typically referred to as epsilon transitions, but in concurrency theory they are represented by a special action $\tau \in Act$. With this, we can then define observable transitions as follows:

$$s \stackrel{\tau}{\Rightarrow} t \quad \text{iff} \quad s \, (\xrightarrow{\tau})^* \, t \quad \text{and}$$
$$s \stackrel{a}{\Rightarrow} t \quad \text{iff} \quad s \, (\xrightarrow{\tau})^* \cdot \xrightarrow{a} \cdot \, (\xrightarrow{\tau})^* \, t \quad \text{for } a \neq \tau.$$

The notion of "behavioural sameness" between two processes (which we view as two states in the same LTS) can be formally captured in many different ways (see, e.g., [6] for an overview). Among those behavioural equivalences, ***bisimilarity*** enjoys special attention. Its formal definition is as follows.

Definition 3. *Let $\mathcal{S} = (S, Act, \rightarrow)$ be an LTS. A binary relation $\mathcal{R} \subseteq S \times S$ is a **bisimulation relation** iff whenever $(s, t) \in R$, we have that*

- *for each transition $s \xrightarrow{a} s'$ there is a transition $t \xrightarrow{a} t'$ such that $(s', t') \in \mathcal{R}$; and*
- *for each transition $t \xrightarrow{a} t'$ there is a transition $s \xrightarrow{a} s'$ such that $(s', t') \in \mathcal{R}$.*

*Processes s and t are **bisimulation equivalent (bisimilar)**, written $s \sim t$, iff they are related by some bisimulation. Thus $\sim$ is the union, and ergo the largest, of all bisimulation relations.*

*If we replace the transition relation $\rightarrow$ in this definition with the weak transition relation $\Rightarrow$, we arrive at the definition of **weak bisimulation equivalence (weak bisimilarity)**, which we denote by $\approx$.*

The above definition of (weak) bisimilarity is a co-inductive one, but can be approximated using the following stratification.

Definition 4. *The **bisimulation approximants** $\sim_\kappa$, for all ordinals $\kappa \in \mathcal{O}$, are defined as follows:*

- $s \sim_0 t$ *for all process states* s *and* t.
- $s \sim_{\kappa+1} t$ *iff*
  - *for each transition* $s \xrightarrow{a} s'$ *there is a transition* $t \xrightarrow{a} t'$ *such that* $s' \sim_\kappa t'$; *and*
  - *for each transition* $t \xrightarrow{a} t'$ *there is a transition* $s \xrightarrow{a} s'$ *such that* $s' \sim_\kappa t'$.
- *For all limit ordinals* λ, $s \sim_\lambda t$ *iff* $s \sim_\kappa t$ *for all* $\kappa < \lambda$.

*The **weak bisimulation approximants** $\approx_\kappa$ are defined by replacing the transition relation $\rightarrow$ with the weak transition relation $\Rightarrow$.*

The following results are then standard.

Theorem 2.

1. *Each* $\sim_\kappa$ *and* $\approx_\kappa$ *is an equivalence relation.*
2. *The relations* $\sim_\kappa$ *and* $\approx_\kappa$ *define strict decreasing hierarchies: over general processes,* $\sim_\kappa \subsetneq \sim_\lambda$ *and* $\approx_\kappa \subsetneq \approx_\lambda$ *whenever* $\kappa > \lambda$.
3. $s \sim t$ *iff* $s \sim_\kappa$ *for all ordinals* $\kappa \in \mathcal{O}$, *and* $s \approx t$ *iff* $s \approx_\kappa t$ *for all ordinals* $\kappa \in \mathcal{O}$. *That is,* $\sim = \cap_{\kappa \in \mathcal{O}} \sim_\kappa$ *and* $\approx = \cap_{\kappa \in \mathcal{O}} \approx_\kappa$.

If $s \not\sim t$, we must have a least ordinal $\kappa \in \mathcal{O}$ such that $s \not\sim_{\kappa+1} t$, and for this value κ we must have $s \sim_\kappa t$. (If $s \not\sim_\lambda t$ for a limit ordinal λ then we must have $s \not\sim_\kappa t$, and hence $s \not\sim_{\kappa+1} t$, for some $\kappa < \lambda$.) We shall identify this value κ by writing $s \sim^!_\kappa t$. In the same way we write $s \approx^!_\kappa t$ to identify the least ordinal $\kappa \in \mathcal{O}$ such that $s \not\approx_{\kappa+1}$.

3.1 Bisimulation Games and Optimal Move Trees

There is a further approach to defining (weak) bisimilarity, one based on games and strategies, whose usefulness is outlined in the tutorial [12]. We describe it here for bisimilarity; its description for weak bisimilarity requires only replacing the transition relation $\rightarrow$ with the weak transition relation $\Rightarrow$, after which all results stated will hold for the weak bisimilarity relations.

A game $\mathcal{G}(s,t)$ corresponding to two states s and t of a process is played between two players, **A** and **B**; the first player **A** (the *adversary*) wants to show that the states s and t are different, while the second player **B** (the *bisimulator)* wants to show that they are the same. To this end the game is played by the two players exchanging moves as follows:

- **A** chooses any transition $s \xrightarrow{a} s'$ or $t \xrightarrow{a} t'$ from one of the states s and t;
- **B** responds by choosing a matching transition $t \xrightarrow{a} t'$ or $s \xrightarrow{a} s'$ from the other state;
- the game then continues from the new position $\mathcal{G}(s', t')$.

The second player **B** wins this game if **B** can match every move that the first player **A** makes (that is, if **A** cannot move or the game continues indefinitely); if, however, **B** at some point cannot match a move made by **A** then player **A** wins. The following is then a straightforward result.

Theorem 3. *$s \sim t$ iff the second player **B** has a winning strategy for $\mathcal{G}(s, t)$.*

If $s \not\sim t$, then $s \sim^!_\kappa t$ for some $\kappa \in \mathcal{O}$, and this κ in a sense determines how long the game must last, assuming both players are playing optimally, before **B** loses the game $\mathcal{G}(s, t)$:

- Since $s \not\sim_{\kappa+1} t$, **A** can make a move such that regardless of **B**'s response the situation will result in a game $\mathcal{G}(s', t')$ in which $s' \not\sim_\kappa t'$; such a move is an ***optimal move*** for **A**.
- For every $\lambda < \kappa$, regardless of the move made by **A**, **B** can respond in such a way that the situation will result in a game $\mathcal{G}(s', t')$ in which $s' \sim_\lambda t'$.

With this in mind, we can make the following definition.

Definition 5. *An **optimal move tree** is a tree whose nodes are labelled by pairs of non-bisimilar states of a process in which an edge $(s, t) \longrightarrow (s', t')$ exists precisely when (s, t) is a node of the tree and the following holds:*

> *In the game $\mathcal{G}(s, t)$, a single exchange of moves in which **A** makes an optimal move may result in the game $\mathcal{G}(s', t')$*

The optimal move tree rooted at (s, t) is denoted by $\boldsymbol{omt}(s, t)$.

If $(s, t) \longrightarrow (s', t')$ is an edge in an optimal move tree, then $s \sim^!_\kappa t$ and $s' \sim^!_\lambda t'$ for some κ and λ with $\kappa > \lambda$. Hence every optimal move tree is well-founded. Furthermore, the following result is easily realised.

Lemma 3. *$h(omt(s, t)) = \kappa$ iff $s \sim^!_\kappa t$.*

3.2 Bounded Branching Processes

Over the class of finite-branching labelled transition systems, it is a standard result that $\sim = \sim_\omega$. We give here a generalisation of this result for infinite-branching processes.

Definition 6. *A limit ordinal κ is **regular** iff it is not the supremum of fewer than κ smaller ordinals.*

Thus for example ω is regular as it is not the supremum of any finite collection of natural numbers. It is easily seen that any regular ordinal must in fact be a cardinal number $\aleph$.

Definition 7. *A process is $<$-$\aleph$-**branching** iff all of its states have fewer than $\aleph$ transitions leading out of them. A tree t is $<$-$\aleph$-**branching** iff all of its nodes have fewer than $\aleph$ children.*

Theorem 4. *If $\aleph$ is a regular cardinal, and t is a well-founded $<$-$\aleph$-branching tree, then $h(t) < \aleph$.*

Proof. By transfinite induction on $h(t)$. If $t \longrightarrow s$ then $h(s) < h(t)$; and by induction $h(s) < \aleph$ and hence $h(s)+1 < \aleph$. Since $h(t) = \sup\{\, h(s)+1 \,:\, t \longrightarrow s \,\}$, by the regularity of $\aleph$ we must have that $h(t) < \aleph$. □

The most basic form of this result is König's Lemma: any finite branching well-founded tree can only have finitely-many nodes (and hence finite height).

The next result follows directly from the fact that $|A \times A| = |A|$ for any infinite set A.

Lemma 4. *If s and t are non-equivalent states of a $<$-$\aleph$-branching process, then $omt(s,t)$ is $<$-$\aleph$-branching.*

¿From the above, we arrive at a theorem on approximant collapse, which generalises the standard result that $\sim = \sim_\omega$ on finitely branching processes as well as a result in [18] concerning the countably-branching processes.

Theorem 5. *For regular cardinals $\aleph$, $\sim = \sim_\aleph$ over the class of $<$-$\aleph$-branching processes.*

Proof. $\sim \subseteq \sim_\aleph$ is a given. If on the other hand $s \not\sim t$, then $h(omt(s,t)) = \kappa$ where $s \sim^!_\kappa t$. Thus, by Lemma 4 and Theorem 4, $\kappa < \aleph$, and hence $s \not\sim_\aleph t$. □

4 Basic Parallel Processes

A Basic Process Algebra (BPA) process is defined by a context-free grammar in Greibach normal form. Formally this is given by a triple $G = (V, A, \Gamma)$, where V is a finite set of *variables* (*nonterminal symbols*), A is a finite set of *labels* (*terminal symbols*), and $\Gamma \subseteq V \times A \times V^*$ is a finite set of *rewrite rules* (*productions*); it is assumed that every variable has at least one associated rewrite rule. Such a grammar gives rise to the LTS $\mathcal{S}_G = (V^*, A, \rightarrow)$ in which the states are sequences of variables, the actions are the labels, and the transition relation is given by the rewrite rules extended by the *prefix rewriting rule*: if $(X, a, \alpha) \in \Gamma$ then $X\beta \xrightarrow{a} \alpha\beta$ for all $\beta \in V^*$. In this way, concatenation of variables naturally represents sequential composition.

A Basic Parallel Processes (BPP) process is defined in exactly the same fashion from such a grammar. However, in this case elements of V^* are read modulo commutativity of concatenation, so that concatenation is interpreted as parallel composition rather than sequential composition. The states of the BPP process associated with a grammar are thus given not by sequences of variables but rather by multisets of variables.

As an example, Figure 1 depicts BPA and BPP processes defined by the same

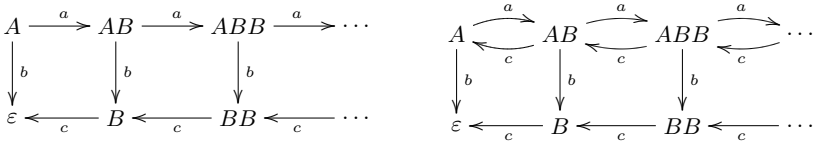

Fig. 1. BPA and BPP processes defined by the grammar $A \xrightarrow{a} AB$, $A \xrightarrow{c} \varepsilon$, $B \xrightarrow{b} \varepsilon$

grammar given by the three rules $A \xrightarrow{a} AB$, $A \xrightarrow{c} \varepsilon$ and $B \xrightarrow{b} \varepsilon$.

Decidability results for (strong) bisimilarity checking have been long established for both BPA [4] and BPP [2, 3]. For a wide class of interest (normed processes) these problems have been shown to have polynomial-time solutions [8–10]. More recently the decision problems for full BPA and BPP have been shown to be PSPACE-hard [14, 15].

Decidability results for weak bisimilarity are much harder to establish, mainly due to the problems of infinite branching. While over BPA and BPP we have $\sim = \cap_{n\in\omega} \sim_m$, the infinite-branching nature of the weak transition relations makes this result false. As an example, Figure 2 gives a BPP process with states P and Q in which $P \approx_n Q$ for all $n \in \omega$ yet $P \not\approx Q$. In this case we have $P \approx_\omega Q$, but from these we can produce BPP process states X_n and Y_n such that $X_n \approx^!_{\omega+n} Y_n$ by adding the following production rules to the defining grammar:

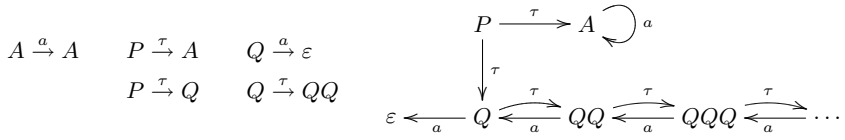

Fig. 2. A BPP process with states P and Q satisfying $P \approx^!_\omega Q$

$$X_1 \xrightarrow{a} P \qquad X_{i+1} \xrightarrow{a} X_i$$
$$Y_1 \xrightarrow{a} Q \qquad Y_{i+1} \xrightarrow{a} Y_i$$

We can construct BPA processes with states X_n and Y_n such that $X_n \approx^!_{\omega^n} Y_n$ [18], but no example BPP states X and Y are known which satisfy $X \approx^!_{\omega\times 2} Y$. This leads to the following long-standing conjecture.

Conjecture 1 (Hirshfeld, Jančar). On BPP processes, $\approx = \approx_{\omega\times 2}$.

BPP processes with silent moves are countably-branching, and thus by Theorem 5 $\approx = \approx_{\aleph_1}$. In [18] there is an argument attributed to J. Bradfield which shows that the approximation hierarchy collapses by $\approx_{\omega_1^{CK}}$, the first non-recursive ordinal. But this is to measure in lightyears what should require centimetres; we proceed here to a more modest bound, based on our ordinal analysis of Dickson's Lemma.

We assume an underlying grammar (V, A, Γ) defining our BPP process, and recall that a state in the associated process is simply a sequence $\alpha \in V^*$ viewed as a multiset. With this, we make the important observation about weak bisimulation approximants over BPP: besides being equivalences, they are in fact congruences.

Lemma 5. *For all $\alpha, \beta, \gamma \in V^*$, if $\alpha \approx_\kappa \beta$ then $\alpha\gamma \approx_\kappa \beta\gamma$.*

Proof. By a simple induction on κ. □

We next observe a result due to Hirshfeld [7].

Lemma 6. *If $\alpha \approx^!_\kappa \beta$ and $\alpha\gamma \approx^!_\mu \beta\delta$ with $\mu<\kappa$, then $\alpha\gamma \approx^!_\mu \alpha\delta$ and $\beta\gamma \approx^!_\mu \beta\delta$.*

Proof. $\alpha\gamma \approx_\mu \alpha\delta$ since $\alpha\gamma \approx_\mu \beta\delta \approx_\kappa \alpha\delta$. On the other hand, if $\alpha\gamma \approx_{\mu+1} \alpha\delta$ then $\alpha\gamma \approx_{\mu+1} \alpha\delta \approx_\kappa \beta\delta$. Thus $\alpha\gamma \approx^!_\mu \alpha\delta$. ($\beta\gamma \approx^!_\mu \beta\delta$ can be shown similarly). □

BPP processes, being multisets over the finite variable set V, can be represented as $|V|$-tuples over $\mathbb{N}$. Given non-equivalent BPP states α and β, $omt(\alpha, \beta)$ can

then be viewed as a $\mathbb{N}^{2\cdot|V|}$-labelled tree. In general this tree will not be non-dominating, but the above lemma will enable us to produce a non-dominating $\mathbb{N}^{2\cdot|V|}$-labelled tree from $omt(\alpha, \beta)$

Lemma 7. *For BPP processes, if $\alpha \approx^!_\kappa \beta$ then there exists a $\mathbb{N}^{2\cdot|V|}$-labelled non-dominating tree of height κ.*

Proof. We apply the following substitution procedure to each successive level of the weak-transition optimal move tree $omt(\alpha, \beta)$ (where the *level* of a node refers to the distance from the root (α, β) to the node):

> for each node u at this level, if u dominates some ancestor node v, that is, if there exists an ancestor node $v = (\varphi, \psi)$ where $u = (\varphi\gamma, \psi\delta)$, then replace the subtree rooted at u with either $u' = omt(\varphi\gamma, \varphi\delta)$ (if $\varphi <_{\text{lex}} \psi$) or $u' = omt(\psi\gamma, \psi\delta)$ (if $\psi <_{\text{lex}} \varphi$). (If this u' itself then dominates an ancestor node, repeat this action.)

That $<_{\text{lex}}$ is a well-founded relation on $\mathbb{N}^{|V|\times 2}$ means this repetition must halt; Lemma 6 implies that this is a height-preserving operation; and the well-foundedness of $omt(\alpha, \beta)$ means that the sequence of levels is finite. □

Theorem 6. *Over BPP processes, $\approx = \approx_{\omega^\omega}$*

Proof. If $\alpha \approx \beta$ then $\alpha \approx_{\omega^\omega} \beta$ is a given. If, on the other hand, $\alpha \not\approx \beta$, then $\alpha \approx^!_\kappa \beta$ for some κ, and by the combination of Lemma 7 and Theorem 1 we must have that $\kappa \leq \omega^{2\cdot|V|}$. Thus, $\alpha \not\approx_{\omega^\omega} \beta$. □

References

1. O. Burkart, D. Caucal, F. Moller and B. Steffen. Verification over Infinite States. Chapter 9 in the **Handbook of Process Algebra**, pp545-623, Elsevier Publishers, 2001.
2. S. Christensen, Y. Hirshfeld, and F. Moller. Bisimulation equivalence is decidable for basic parallel processes. In *Proceedings of the 4th International Conference on Concurrency Theory (CONCUR'93)*, LNCS 715, pages 143-157, Springer, 1993.
3. S. Christensen, Y. Hirshfeld, and F. Moller. Decomposability, decidability and axiomatisability for bisimulation equivalence on basic parallel processes. In *Proceedings of the 8th Annual IEEE Symposium on Logic in Computer Science (LICS'93)*, pages 386-396, IEEE Computer Society Press, 1993.
4. S. Christensen, H. Hüttel, and C. Stirling. Bisimulation equivalence is decidable for all context-free processes. In *Proceedings of the 3rd International Conference on Concurrency Theory (CONCUR'92)*, LNCS 630, pages 138-147, Springer, 1992.

5. L.E. Dickson. Finiteness of the odd perfect and primitive abundant numbers with distinct factors. *American Journal of Mathematics* **3**:413-422, 1913.
6. R. Glabbeek. The linear time – branching time spectrum I: The semantics of concrete sequential processes. Chapter 1 in the **Handbook of Process Algebra**, pp3-99, Elsevier Publishers, 2001.
7. Y. Hirshfeld. Bisimulation trees and the decidability of weak bisimulation. *Electronic Notes in Theoretical Computer Science* **5**:2-13, 1997.
8. Y. Hirshfeld, M. Jerrum, and F. Moller. A polynomial algorithm for deciding bisimilarity of normed context-free processes. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science (FOCS'94)*, pages 623-631, IEEE Computer Society Press, 1994.
9. Y. Hirshfeld, M. Jerrum, and F. Moller. A polynomial algorithm for deciding bisimilarity of normed context-free processes. *Theoretical Computer Science* **15**:143-159, 1996.
10. Y. Hirshfeld, M. Jerrum, and F. Moller. A polynomial algorithm for deciding bisimulation equivalence of normed basic parallel processes. *Mathematical Structures in Computer Science* **6**:251-259, 1996.
11. 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)*, pages 218-227, IEEE Computer Society Press, 2003.
12. P. Jančar and F. Moller. Techniques for decidability and undecidability of bisimilarity. In *Proceedings of the 10th International Conference on Concurrency Theory (CONCUR'99)*, LNCS 1664, pages 30-45, Springer, 1999.
13. F. Moller. Infinite Results. In *Proceedings of the 7th International Conference on Concurrency Theory (CONCUR'96)*, LNCS 1119, pages 195-216, Springer, 1996.
14. J. Srba. Strong bisimilarity and regularity of Basic Process Algebra is PSPACE-hard. In *Proceedings of the 29th International Conference on Automata, Languages and Programming (ICALP'02)*, LNCS 2380, pages 716-727, Springer, 2002.
15. J. Srba. Strong bisimilarity and regularity of Basic Parallel Processes is PSPACE-hard. In *Proceedings of the 19th International Symposium on Theoretical Aspects of Computer Science (STACS'02)*, LNCS 2285, pages 535-546, Springer, 2002.
16. J. Srba. Complexity of weak bisimilarity and regularity for BPA and BPP. *Mathematical Structures in Computer Science* **13**:567-587, 2003.
17. J. Srba. Roadmap of Infinite Results. http://www.brics.dk/~srba/roadmap.
18. J. Stribrna. *Decidability and complexity of equivalences for simple process algebras*. PhD thesis ECS-LFCS-99-408, University of Edinburgh, 1999.