URL: http://www.elsevier.nl/locate/entcs/volume27.html 22 pages

Decidability of Plain and Hereditary History-Preserving Bisimilarity for BPP

Sibylle Fröschle

LFCS, University of Edinburgh
Scotland
sib@dcs.ed.ac.uk

Abstract

In this paper we investigate the decidability of history-preserving bisimilarity (HPB) and hereditary history-preserving bisimilarity (HHPB) for basic parallel processes (BPP). We find that both notions are decidable for this class of infinite systems, and present tableau-based decision procedures. The first result is not new but has already been established via the decidability of causal bisimilarity, a notion that is equivalent to HPB. We shall see that our decision procedure is similar to Christensen's proof of the decidability of distributed bisimilarity, which leads us to the coincidence between HPB and distributed bisimilarity for BPP. The decidability of HHPB is a new result. This result is especially interesting, since the decidability of HHPB for finite-state systems has been a long-standing open problem which has recently been shown to be undecidable.

1 Introduction

One important problem in the verification of concurrent systems is to check whether two given systems E and F are equivalent under a given notion of behavioural equivalence. In the world of finite-state systems this verification problem is decidable for the standard equivalences, since one can theoretically proceed by exhaustive search. The challenge lies then in finding algorithms of low complexity.

For infinite-state systems the equivalence problem cannot be decidable in general, due to the theoretical limits set by the halting problem. However, restricted classes of infinite systems have been defined and investigated, and in the interleaving world many interesting and often surprising results have already been established [23]. For example, in [9] it has been shown that classical bisimilarity is decidable for the class of basic parallel processes (BPP).

[1] This work has been supported by CONFER-2 (Concurrency and Functions: Evaluation and Reduction-2), working group 21836. Part of the work was done while the author was at BRICS, University of Aarhus, Denmark.

©*1999 Published by Elsevier Science B. V.* Open access under CC BY-NC-ND license.

BPP can be seen as an extension of finite automata by a parallel operator, and therefore they are the natural system class to consider when exploring non-interleaving semantics in the infinite-state world. Several results have already been found. One of the earliest such results is Christensen's proof of the decidability of distributed bisimilarity for BPP and BPP_τ, the extension of BPP with communication [7,8]. In [22] Kiehn and Hennessy present decision procedures for strong and weak versions of causal bisimulation, location equivalence and ST-bisimulation, also for the system class of BPP_τ. A result for linear-time equivalences has been established in [27], where Sunesen and Nielsen prove the decidability of a causality- and locality-based trace equivalence for BPP.

Interestingly, several important non-interleaving equivalences coincide for BPP-like process languages. In [2] Aceto shows that distributed, timed and causal bisimulation coincide for a language that is essentially BPP without recursion. Kiehn has recently extended these results by proving that location equivalence, causal bisimulations and distributed bisimulations coincide over CPP, a language that corresponds to BPP_τ without explicit τ actions [21].

In this paper we investigate history-preserving bisimulation (HPB) [26,10,28,4] and hereditary history-preserving bisimulation (HHPB)[3,18], and find that both notions are decidable for BPP. The decidability for HPB is not a new result. It is already established via the decidability of causal and distributed bisimulation, since in [1] it is proved that HPB and causal bisimulation coincide for stable event structures.

We shall see that our tableau system for HPB is very similar to the one Christensen employs in his proof of the decidability of distributed bisimilarity [7]. Indeed our tableau establishes the decidability of distributed bisimilarity just as well, which immediately gives us an alternative proof of the coincidence of history-preserving and distributed bisimilarity for BPP.

The decidability of HHPB for BPP is a new result, and constitutes the main contribution of this paper. It has been a long-standing open problem whether HHPB is decidable for finite-state systems. Some results approaching the problem are presented in [15] and [20], but interestingly, the problem has recently been proved to be undecidable in [19]. Thus with HHPB we have an equivalence that is undecidable for finite-state systems but decidable for a class of infinite-state systems. In contrast HPB for finite-state systems has been shown to be decidable in [29,17,24], and its weak version in [30,17].

When considering plain and hereditary HPB for BPP, first of all we need to define partial order semantics for BPP. We do this in Sec. 2, where we first introduce a new notion of normal form for BPP, on which we then build our partial order semantics. Sec. 3 gives the tableau-based decision procedure for HPB and in Sec. 4 we state the coincidence result. The decision procedure for HHPB follows in Sec. 5. Sec. 6 gives directions for further investigations.

2 Basic Parallel Processes

We start with the definition of basic parallel processes (BPP) following [8].

Definition 2.1 Let $Act = \{a, b, c, \ldots\}$ be a countably infinite set of atomic actions, and let $Var = \{X, Y, Z, \ldots\}$ be a countably infinite set of process variables. *BPP process expressions* are given by the following grammar:

$$\begin{aligned} E ::= & \ \mathbf{0} && \text{(inaction)} \\ | & \ X && \text{(process variable, } X \in Var\text{)} \\ | & \ a.E && \text{(action prefix, } a \in Act\text{)} \\ | & \ E + E && \text{(choice)} \\ | & \ E \,||\, E && \text{(parallel merge)} \end{aligned}$$

A *BPP* $\mathcal{E}$ is a finite family of recursive process equations $\mathcal{E} = \{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}$, where the X_i are distinct variables and the E_i are BPP process expressions only containing variables of the set $Var(\mathcal{E}) = \{X_1, X_2, \ldots, X_n\}$.

We define the variable X_1 to be the *leading variable* of $\mathcal{E}$, and $X_1 \stackrel{def}{=} E_1$ to be the *leading equation* of $\mathcal{E}$, correspondingly.

A process expression E is *guarded* iff every variable in E occurs within the scope of action prefix. We say a BPP $\mathcal{E} = \{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}$ is *guarded* iff each E_i is guarded. In the following we shall only consider guarded BPP.

In the interleaving world one usually considers BPP in so-called full standard form only. The characteristic of a BPP $\mathcal{E}$ in full standard form is that every defining expression E_i is of the form $\sum_{j=1}^{n_i} a_{ij}.\alpha_{ij}$, where each α_{ij} is a parallel merge of variables of $\mathcal{E}$. The concept originates from [8], and there it has also been shown that every BPP can be effectively transformed into a bisimilar BPP in full standard form. Hence, in the interleaving world it is indeed justified to deal with such BPPs only. This is not valid for the truly-concurrent world, since the transformation relies on the validity of the expansion law. To handle BPP efficiently under partial order semantics, we need to develop a new kind of normal form.

Execution Normal Form.

I suggest a very simple normal form, the so-called execution normal form (ENF). A BPP is in ENF if in every defining expression every variable occurrence is immediately guarded and every action prefix is directly followed by a variable. Hence, a ENF process expression is based on subexpressions of the form $a.X$ or $\mathbf{0}$, which are arbitrarily nested by choice and parallel merge.

We call this normal form "execution normal form" because all the actions occurring in an ENF expression are enabled, and so one can easily read from an expression which actions can be executed next. As we shall see later in our proofs, ENF is especially suited for partial order semantics.

Definition 2.2 The class of BPP expressions in *execution normal form* (short: *ENF expressions*) is defined by the following grammar

$$E ::= \mathbf{0} \mid a.X \mid E + E \mid E \mid\mid E.$$

A BPP $\mathcal{E} = \{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}$ is defined to be in *ENF* iff every expression E_i is in ENF.

Execution normal form is very unrestricted in that every BPP can effectively be transformed into a BPP in ENF by using operations, that only affect the appearance of the set of defining equations and do not rely on any semantic laws.

Lemma 2.3 *Every BPP can effectively be transformed into a BPP in ENF by a sequence of operations that only affect the appearance of the set of defining equations.*

Proof. The only operations we need to transform a BPP into a BPP in ENF are: introduction of new variables, substitution of new variables for subexpressions, and unfolding of variables. Since we consider only guarded BPP the termination of the transformation is guaranteed. □

It is clear that any semantic equivalence of interest is preserved under such operations. Hence, when in the following we restrict ourselves to BPP in ENF, we still cover the whole class of BPP.

BPP in ENF with Labelling.

We would like to distinguish between different occurrences of the same action in an ENF expression. We can do this by employing labelled transitions [2] instead of actions.

Definition 2.4 Let T be a countably infinite set of transitions. We redefine *ENF expressions* by the grammar

$$E ::= \mathbf{0} \mid t.X \mid E + E \mid E \mid\mid E,$$

where $t \in T$.

We define the *transitions* of an ENF expression E as $T(E) = \{t \mid \text{ for some } X, t.X \text{ is a subexpression of } E\}$.

We redefine a *BPP in ENF* $\mathcal{E}$ to be a pair $(\Delta_\mathcal{E}, l_\mathcal{E})$, where $\Delta_\mathcal{E} = \{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}$ is a finite family of process equations such that every E_i is in ENF, and for each t there is at most one subexpression of the form "$t.X$" in the E_i. $l_\mathcal{E} : \bigcup_{E_i} T(E_i) \to Act$ is the labelling function.

Let E be a ENF expression. We define the *base expressions* of E to be the set $BaseE(E) = \{E' \mid E' = \mathbf{0}$ or $E' = t.X,$ and E' is a subexpression of E$\}$.

From now on we only consider BPP in ENF with labelling. It is clear that each BPP can easily be given as such a BPP, just as well.

[2] We use the name "transitions" following Petri net terminology.

We shall often assume the family of defining equations and the labelling function to be given implicitly. Thus, whenever we speak of an ENF expression E, we assume E to be a defining expression of an implicit BPP in ENF. Note that this guarantees that the transitions of E are distinct.

In the following we sometimes regard an expression E as a BPP, namely the one defined by the underlying implicit BPP with the leading expression set to E. We usually use E and F to denote ENF expressions, and $\mathcal{E}$ and $\mathcal{F}$ to denote the underlying BPPs.

As another convention we shall identify BPP expressions up to structural congruence, i.e. associativity, commutativity, and $\mathbf{0}$ absorption of choice and parallel merge.

2.1 Partial Order Semantics for BPP

We now give partial order semantics to BPP. Similar to [13] we do this by translating every BPP in ENF into a labelled occurrence net, which gives us the unfolding of the BPP. In [13] BPPs are first translated into a P/T net representation, which is then transformed into its unfolding via the standard partial order semantics for P/T nets given in [11]. Here we give a direct translation from BPP into occurrence nets since it is not clear what the correct P/T net representation for BPP in ENF would be (compare section 6).

Let us first introduce the necessary Petri net terminology.

Petri net terminology.

Most of our definitions follow [13], sometimes they are slightly changed reflecting that we only consider nets without weights.

A labelled net N is a tuple (S_N, T_N, F_N, l_N), where S_N is the set of places, T_N is the set of transitions, $F_N : (S_N \times T_N) \cup (T_N \times S_N) \to \{0, 1\}$ is the flow relation, and $l_N : S_N \cup T_N \to Act$ is the labelling function, where Act is a set of actions.

The pre-set of an element $x \in S_N \cup T_N$, ${}^\bullet x$, is defined by $\{y \mid F_N(y, x) > 0\}$, the post-set of x, $x^\bullet$, similarly is $\{y \mid F_N(x, y) > 0\}$.

A marking M of N is a map $S_N \to \mathbb{N}_0$. M enables a transition $t \in T_N$ if $M(s) \geq F(s, t)$ for every $s \in S_N$. If t is enabled at M it can occur. The resulting marking M' is defined by $M'(s) = M(s) - F(s, t) + F(t, s)$ for all $s \in S_N$. We denote this by $M \xrightarrow{t} M'$. Similarly, we write $M \xrightarrow{w} M'$, if there exist a sequence $w = t_1 \dots t_n$ and markings $M_1, M_2, \dots, M_{n-1}, M'$ such that $M \xrightarrow{t_1} M_1 \xrightarrow{t_2} M_2 \dots M_{n-1} \xrightarrow{t_n} M'$, or if $w = \epsilon$, and $M' = M$. We use similar notation when we want to hide the transitions behind the action labels.

Let N be a net, and $init_N$ be a marking of N. Then the pair $(N, init_N)$ is a *Petri net* with initial marking $init_N$. A marking M is *reachable* in $PN = (N, init_N)$ if we have some w such that $init_N \xrightarrow{w} M$. We denote the set of reachable markings of PN by $Reachable(PN)$.

PN is *1-safe* if for every marking $M \in Reachable(PN)$ we have: $M(s) \leq 1$ for every $s \in S_N$. Thus, in 1-safe nets a marking can be viewed as a set of places.

Let (S_N, T_N, F_N, l_N) be a net and let $x_1, x_2 \in S_N \cup T_N$. We say x_1 and x_2 are in *conflict*, denoted by $x_1 \# x_2$, if there exist distinct transitions $t_1, t_2 \in T_N$ such that ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$, and there exist paths in the net leading from t_1 to x_1, and from t_2 to x_2. $x \in S_N \cup T_N$ is in *self-conflict* iff $x \# x$. Note that this definition of conflict is not intuitive for general nets, but in the context of our semantics we think of 1-safe occurrence nets.

A *labelled occurrence net*[3] is an acyclic net $N = (B_N, E_N, F_N, l_N)$ such that:

- for every $b \in B_N$, $|{}^\bullet b| \leq 1$,
- N is finitely preceded, i. e., for every $x \in B_N \cup E_N$, the set of elements $y \in B_N \cup E_N$ such that there exists a path from y to x is finite,
- no $e \in E_N$ is in self-conflict.

Note that we call the places of occurrence nets *conditions*, and the transitions *events*. For two elements $x, y \in B_N \cup E_N$ we define $x \leq_N y$ iff there exists a path from x to y. Since occurrence nets are acyclic, it is clear that $\leq_N$ is a partial order. We denote the set of minimal elements of $\leq_N$ by ${}^\circ N$, and the set of maximal elements by N°, respectively.

Our translation from BPP to occurrence nets uses two types of composition for nets, the *parallel merge* of two nets, and the *choice composition* of two nets.

Definition 2.5 For both constructions we need the *disjoint union* of two nets N_1, N_2. It is defined by

$$\begin{aligned} N_1 \uplus N_2 \stackrel{def}{=} (&S_{N_1} \uplus S_{N_2}, T_{N_1} \uplus T_{N_2}, \\ &\{((x,i),(y,i)) \mid (x,y) \in F_{N_i} \text{ for } i \text{ either 1 or 2}\}, \\ &\{((p,i), l_{N_i}(p)) \mid i \text{ either 1 or 2}\}). \end{aligned}$$

In the *parallel merge* of two nets we simply juxtapose the nets. This amounts to the disjoint union, and so we define: $N_1 \mid\mid N_2 \stackrel{def}{=} N_1 \uplus N_2$.

We define the *choice composition* only for acyclic nets. The idea is that the Petri net $(N_1 + N_2, {}^\circ(N_1 + N_2))$ either behaves like $(N_1, {}^\circ N_1)$ or like $(N_2, {}^\circ N_2)$ depending on the choice of the first transition. If the nets are labelled we require that the labels of ${}^\circ N_1 \cup {}^\circ N_2$ are all identical. The choice composition is then given by

$$\begin{aligned} &N_1 + N_2 \stackrel{def}{=} \\ &\quad ((C_{N_1 \uplus N_2} - {}^\circ(N_1 \uplus N_2)) \cup \{(p_1,1) \mid p_1 \in {}^\circ N_1\} \times \{(p_2,2) \mid p_2 \in {}^\circ N_2\}, \\ &\quad E_{N_1 \uplus N_2}, \end{aligned}$$

[3] We use occurrence nets in the sense of [25] allowing forwards conflict.

$(F_{N_1 \uplus N_2} - \{(p,e) \mid p \in {}^\circ(N_1 \uplus N_2)\}) \cup$
$\{(((p_1,1),(p_2,2)),e) \mid ((p_1,1),e) \in F_{N_1 \uplus N_2} \text{ or } ((p_2,2),e) \in F_{N_1 \uplus N_2}\},$
$(l_{N_1 \uplus N_2} - \{(p,a) \mid p \in {}^\circ(N_1 \uplus N_2)\}) \cup \{(((p_1,1),(p_2,2)),a) \mid (p_1,a) \in l_{N_1}\}).$

Figure 1 shows the algorithm that translates a BPP $\mathcal{E}$ into its unfolding $Unf(\mathcal{E})$. The core of the transformation is the function $unf1X$. It unfolds a given BPP variable X by one level; more precisely, it recursively generates a net fragment that represents the defining ENF expression E of X. The events of this net fragment correspond to the transitions of E, and thereby to the base expressions of E. The event corresponding to the base expression $t.X_i$ has one postcondition labelled by X_i. The preplaces of the events reflect the nesting of choice and parallel merge of their corresponding base expressions within E.

The function $unfold$ uses $unf1X$ to extend a partial unfolding N in breadth-first manner level by level, i. e. it unfolds each condition of N° by one level, and then calls itself recursively.

Usually the unfolding of a BPP $\mathcal{E}$ will be infinite, and our transformation will not terminate. Then we define $Unf(\mathcal{E})$ to be the obvious infinite object. More precisely, we could have defined a notion of branching processes for BPP, shown that the set of all branching processes of a given BPP $\mathcal{E}$ forms a complete lattice, and then defined $Unf(\mathcal{E})$ as the maximal element of the lattice (Analogously to Engelfriet's definition of the unfolding of a P/T net in [11].).

Petri Net Representation.

Note that $Unf(\mathcal{E})$ is indeed a labelled occurrence net. If we equip this net with a suitable initial marking, we gain a Petri net representation for BPP based on possibly infinitary 1-safe Petri nets. Formally, we define the representation of a BPP $\mathcal{E}$ as a 1-safe Petri net by:

$$PN(\mathcal{E}) = (Unf(\mathcal{E}), {}^\circ Unf(\mathcal{E})).$$

This characterization is interesting since the well-known one-to-one correspondence between BPP and communication-free Petri nets [12] relies on the existence of a full standard normal form for every BPP, and is therefore only valid in the interleaving world. In [13] following [16] it has been suggested that there is still a way to represent BPP as communication-free nets: one can introduce silent transitions to model the nesting of choice and parallel merge.

Every reachable marking of $PN(\mathcal{E})$ corresponds to an ENF expression E, which can easily be determined by looking at the net and the defining equations of $\mathcal{E}$. On the other hand, every ENF expression E can be viewed as the marking ${}^\circ Unf(E)$. In the following we will make use of these correspondences; sometimes we consider an expression as a marking, or view a marking as a process expression.

BPP Processes.

Having defined the unfolding of a BPP, it is straightforward to define a notion of partial order run for BPP. Similar to [13] we simply take subnets of $Unf(\mathcal{E})$

```
newC() -- get a new unused condition

newE() -- get a new unused event

unf1X(X:  a variable of ℰ)
      -- unfold the variable X of ℰ by one level

      transExpr(E:  a ENF expression)
         -- translate the ENF expression E into a net fragment,
         -- we assume that the net composition functions
         -- use newC and newE so as to avoid conflicting ids.
         case E of
         E = 0  => ({c = newC()}, {}, {}, {(c, 0)})
         E = t.X_i  => ({c_1 = newC(), c_2 = newC()}, {e = newE()},
             {(c_1, e), (e, c_2)}, {(c_1, X), (c_2, X_i), (e, t)})
         E = E_1 || E_2  => transExpr(E_1) || transExpr(E_2)
         E = E_1 + E_2  => transExpr(E_1) + transExpr(E_2)
      end transExpr

      transExpr(E)
         where E is the defining expression of X.
end unf1X

unfold(N:  a partial unfolding)
      -- further unfold a partial unfolding N, breadth first
      forall {p_i | p_i ∈ N° ∧ l_N(p_i) ≠ 0} do
         -- unfold p_i
         N_{p_i} := unf1X(l_N(p_i))

         -- now substitute N_{p_i} for p_i in N
         If |•p_i| = 0 then
             N := N_{p_i}
         else
             N := ((C_N − p_i) ∪ C_{N_{p_i}}, E_N ∪ E_{N_{p_i}},
                 (F_N − (e, p_i)) ∪ {(e, p) | p ∈ C_{N_{p_i}}} ∪ F_{N_{p_i}}, (l_N − (p_i, l_N(p_i))) ∪ l_{N_{p_i}})
                 where e is the one event such that e ∈ •p_i.

      If |{p | p ∈ N° ∧ l_N(p) ≠ 0}| > 0 then
         unfold(N)
end unfold

Unf(ℰ: a BPP in ENF)
      startnet := ({c = newC()}, {}, {}, {(c, X_1)})
      unfold(startnet)
end Unf
```

Fig. 1. Unf($\mathcal{E}$)

satisfying special conditions. Formally, we define a process π of a BPP $\mathcal{E}$ to be a finite subnet of $Unf(\mathcal{E})$, such that ${}^\circ\pi = {}^\circ Unf(\mathcal{E})$, and every condition of π has at most one output event in π. We consider processes only up to isomorphism.

We define the initial process of a BPP $\mathcal{E}$ by $\pi_0(\mathcal{E}) = ({}^\circ Unf(\mathcal{E}), \{\}, \{\}, l_{Unf(\mathcal{E})} \restriction_{{}^\circ Unf(\mathcal{E})})$.

Let E be an ENF expression and $\sigma = \{t_1, \ldots, t_n\} \subseteq T(E)$, such that for some E', $E \xrightarrow{w} E'$, where $w = t_1 \ldots t_n$. Then we denote the process of E that corresponds to the occurrence of σ by $pr(E, \sigma)$. If we want the result to be a process of ${}^\bullet\sigma$ we write $pr^-(E, \sigma)$. When σ is a singleton $\{t\}$, we write $pr(E, t)$, and $pr^-(E, t)$, respectively .

If we have a process π such that $\pi^\circ \xrightarrow{t} M'$ for some t, M', we write $\pi \xrightarrow{t} \pi'$, where π' is the process extended by the occurrence of t.

Let π_1 and π_2 be two processes such that π_1° and ${}^\circ\pi_2$ correspond to the same ENF expression. Then we can compose π_1 and π_2 in the obvious way. We denote the result by $\pi_1 \circ \pi_2$.

In the context of a process π, we shall often refer to the events corresponding to a set of transitions by the transitions themselves, if this can be done without causing ambiguity.

2.2 Plain and Hereditary History-Preserving Bisimulation

For the definition of plain and hereditary HPB we need even more definitions. The pomset definitions follow the presentation in [17].

Pomsets.

A *pomset* is a labelled partial order. It is a tuple $p = (E_p, <_p, L_p, l_p)$, where E_p is a set of events, $<_p$ a partial order relation on E_p, L_p is a set of labels, and l_p a labelling function $l_p : E_p \to L_p$.

A function f is an $isomorphism$ between pomset p and pomset q iff $f : E_p \to E_q$ is a bijection, such that we have $l_p = l_q \circ f$, and $e <_p e'$ iff $f(e) <_q f(e')$ for all $e, e' \in E_p$.

Let π be a process of a BPP $\mathcal{E}$. The *pomset* of π is defined as $pom(\pi) = (E_\pi, \leq_\pi \restriction_{E_\pi \times E_\pi}, Image(l_\mathcal{E} \circ (l_\pi \restriction_{E_\pi})), l_\mathcal{E} \circ (l_\pi \restriction_{E_\pi}))$.

Conventions.

For a triple (π_E, π_F, f) where π_E is a process of an ENF expression E, π_F a process of an ENF expression F, and f a pomset isomorphism between $pom(\pi_E)$ and $pom(\pi_F)$, we use π as short notation. In the context of such a triple π, we write t for a pair of transitions (t_E, t_F) when we have $f(e_{t_E}) = e_{t_F}$, where e_{t_E}, e_{t_F} denote the events corresponding to the last occurrences of t_E, t_F in $pom(\pi_E)$, $pom(\pi_F)$.

Further we use $\pi \subseteq \pi'$ when we have two triples π, π' such that $\pi_E \subseteq \pi'_E$, $\pi_F \subseteq \pi'_F$, and $f = f' \restriction_{pom(\pi_E)}$.

Similarly, whenever we have a set $\sigma \in \mathcal{P}(T(E) \times T(F))$ we use σ_E to denote the set $\{t_E \mid (t_E, t_F) \in \sigma \text{ for some } t_F\}$, and we use σ_F analogously.

We employ analogous conventions for the net compositions '$||$' and '$\circ$', and the functions pr and pr^-.

Definition of Plain and Hereditary History-Preserving Bisimulation.

Here comes the definition of plain and hereditary history-preserving bisimulation.

Definition 2.6 Let E, F be ENF expressions. A set $\mathcal{H}$ of triples (π_E, π_F, f) is a *history-preserving bisimulation* for E, F if

(i) Whenever $(\pi_E, \pi_F, f) \in \mathcal{H}$, then π_E is a process of E, π_F is a process of F and f is a pomset isomorphism from $pom(\pi_E)$ onto $pom(\pi_F)$.

(ii) $(\pi_0(E), \pi_0(F), \emptyset) \in \mathcal{H}$.

(iii) Whenever $(\pi_E, \pi_F, f) \in \mathcal{H}$ and $\pi_E \xrightarrow{t_E} \pi'_E$ for some t_E, π'_E, then there exist t_F, π'_F, f' such that $\pi_F \xrightarrow{t_F} \pi'_F$, $(\pi'_E, \pi'_F, f') \in \mathcal{H}$ and $f' \restriction_{pom(\pi_E)} = f$.

(iv) Vice versa.

A history-preserving bisimulation is *hereditary* when it further satisfies

(v) Whenever $\pi \in \mathcal{H}$ and $\pi' \subseteq \pi$ for some π', then $\pi' \in \mathcal{H}$.

Two ENF expressions E and F are *(hereditary) history-preserving bisimilar*, written $E \sim_{(H)HPB} F$, iff there is a (hereditary) history-preserving bisimulation relating them.

3 Decidability of History-Preserving Bisimilarity for BPP

Let us begin with the introduction of a concept that is crucial for the proof. In [5] Castellani introduces a non-interleaving semantics based on the principle of distribution. To reflect that concurrent processes are situated at different locations, a transition in her distributed transition systems leads to a compound residual, consisting of a local residual and a concurrent residual. The local residual describes the remaining behaviour of the locality where the action took place, whereas the concurrent residual represents the unaffected behaviour of the localities that have not been involved in the action performance. The parallel composition of the two residuals constitutes the global remaining behaviour.

In our partial order semantics we can split the system behaviour that remains after the execution of an action a into two parallel components just as well. Then one component describes the remaining behaviour that is dependent on a, whereas the other stands for the remaining behaviour independent of a. We call these components the local and parallel remainder of the corresponding expression E. For ENF expressions we can define these entities with respect to a transition t as follows.

Definition 3.1 Let E be an ENF expression, and let t be a transition enabled at E,

that is $t.X \in BaseE(E)$ for some variable X. We define the *local remainder* of E after the occurrence of transition t by

$$localR(E,t) = X,$$

and the *parallel remainder* of E after the occurrence of transition t inductively by

$$\begin{aligned}
parallelR(t.X,t) &= \mathbf{0}\\
parallelR(E_1 \mid\mid E_2,t) &= \textbf{if } t.X \in E_1 \textbf{ then } parallelR(E_1,t) \mid\mid E_2\\
&\quad\ \textbf{else } E_1 \mid\mid parallelR(E_2,t)\\
parallelR(E_1 + E_2,t) &= \textbf{if } t.X \in E_1 \textbf{ then } parallelR(E_1,t)\\
&\quad\ \textbf{else } parallelR(E_2,t)
\end{aligned}$$

Observe that the outcome of $parallelR$ is always an expression in ENF, since this function merely filters out some expressions according to the choice structure of the argument.

Notation 1 *Let E be an ENF expression, and let t be a transition enabled at E, that is $t \in T(E)$. We write $E \stackrel{t}{\rightarrow} (E_l, E_p)$ as a short notation whenever we have $localR(E,t) = E_l$ and $parallelR(E,t) = E_p$ (where $=$ is up to structural congruence). Note that the process $E_l \mid\mid E_p$ corresponds exactly to the process E', where E' is given by $E \stackrel{t}{\rightarrow} E'$.*

We establish the decidability of HPB for BPP by means of a tableau system. As we shall see our proof is very similar to [7] where the tableau technique has been employed to establish the decidability of distributed bisimilarity for BPP.

A tableau system is a goal-directed proof scheme: to prove that two given systems $\mathcal{E} = (\{X_i \stackrel{def}{=} E_i : i = 1,2,\ldots,n\}, l_{\mathcal{E}})$ and $\mathcal{F} = (\{Y_j \stackrel{def}{=} F_j : j = 1,2,\ldots,m\}, l_{\mathcal{F}})$ are equivalent one starts with the goal $X_1 = Y_1$ and builds from this root node a proof tree. This is done by applying proof rules to obtain subgoals according to the structure of the expressions. The proof rules are in turn applied to the subgoals, and this process is repeated until a node is recognized as a terminal node. Terminal nodes can either be successful or unsuccessful. We say a tableau is successful iff it is finite and all its terminals are successful.

Let us introduce some standard tableau terminology as it can be found e. g. in [9]. We denote a tableau with the root labelled "$X = Y$" by $T(X = Y)$. We use the letter π to designate paths through a tableau, and the letter n to denote nodes of a tableau. When we want to indicate the label of a node n we write $n : E = F$.

Now we present our tableau system for HPB. Figure 2 gives the proof rules. Note that our rules only cover goals of the form "$X_i = Y_j$" or "$E = F$", where E and F are ENF expressions. This is sufficient since we start the tableau with a node of the first form, and our rules only generate subgoals of either form. This is obvious for **Rec**, and the first half of subgoals of **Match**. It also becomes clear for the second half of subgoals when one remembers that $parallelR$ only outputs expressions in ENF.

$$\textbf{Rec}\ \frac{X_i = Y_j}{E_i = F_j}$$

$$\textbf{Match}\ E = F \Leftarrow \begin{cases} \{localR(E, t_E) = localR(F, f(t_E))\}_{\forall t_E \in T(E)} \\ \{localR(E, g(t_F)) = localR(F, t_F)\}_{\forall t_F \in T(F)} \\ \{parallelR(E, t_E) = parallelR(F, f(t_E))\}_{\forall t_E \in T(E)} \\ \{parallelR(E, g(t_F)) = parallelR(F, t_F)\}_{\forall t_F \in T(F)} \end{cases}$$

where $f : T(E) \to T(F)$

$g : T(F) \to T(E)$

such that the functions g and f are label-preserving,

i. e. forall $t_E \in T(E)$ we have $l_{\mathcal{E}}(t_E) = l_{\mathcal{F}}(f(t_E))$,

and similar for g.

Fig. 2. Tableau Rules for HPB

The terminal conditions of our tableau system are as follows. A node $n : label$ is a *successful terminal* if one of the following conditions holds:

(i) $label =$ "$\mathbf{0} = \mathbf{0}$".

(ii) $label =$ "$X = Y$", and there is an ancestor node n_a above n in the tableau such that n_a is labelled with "$X = Y$" as well.

A node $n : label$ is an *unsuccessful terminal* if the following condition holds:

(i) $label =$ "$E = F$", where E and F are ENF expressions, and a pair of functions f and g as required by rule **Match** does not exist.

It is no problem to check the latter condition. Since we only deal with finite-branching systems we can simply do so by exhaustive search.

In the following we prove finiteness, completeness and soundness for our tableau system. Altogether this will establish the decidability of HPB.

Lemma 3.2 (Finiteness) *Every tableau for two given BPP systems is finite. Furthermore, for two given BPP systems the number of possible tableaux is finite.*

Proof. Let $\mathcal{E} = (\{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}, l_{\mathcal{E}})$ and $\mathcal{F} = (\{Y_j \stackrel{def}{=} F_j : j = 1, 2, \ldots, m\}, l_{\mathcal{F}})$ be two given BPP in ENF. Assume to the contrary an infinite tableau $T(X_1 = Y_1)$. Since we consider only guarded BPP any tableau will be finite-branching, so we can apply König's lemma and assume an infinite path π through the tableau. It is easy to see that any infinite path must contain an infinite number of instantiations of the rule **Rec**. But this immediately leads to a contradiction. There are only n variables in $\mathcal{E}$ and m variables in $\mathcal{F}$. Thus, we only have

$m \times n$ different nodes of the form $X_i = Y_j$ at our disposal, and after at most $m \times n$ instances of **Rec** we hit a terminal node by the second condition for successful terminals.

This observation also establishes an upper bound on every tableau for given $\mathcal{E}$ and $\mathcal{F}$. Thus, clearly there can be only finitely many different tableaux. □

Before we proceed to the proof of completeness we need to establish the following essential lemma.

Lemma 3.3 (Forward Soundness of Match) *Let E and F be two ENF process expressions such that $E \sim_{HPB} F$. Then the following property holds,*

- *Whenever $E \xrightarrow{t_E} (E_l, E_p)$, then there exist t_F, (F_l, F_p) such that we have $l_{\mathcal{E}}(t_E) = l_{\mathcal{F}}(t_F)$, $F \xrightarrow{t_F} (F_l, F_p)$, $E_l \sim_{HPB} F_l$ and $E_p \sim_{HPB} F_p$.*
- *Vice versa.*

Proof. Assume $\mathcal{H}$ to be a HPB relating E and F. By definition of HPB, whenever we have $E \xrightarrow{t_E} (E_l, E_p)$, then there is $F \xrightarrow{t_F} (F_l, F_p)$ such that $(\pi_E = pr(E, t_E), \pi_F = pr(F, t_F), f = \{(t_E, t_F)\}) \in \mathcal{H}$. Since f is a pomset isomorphism, it is clear that $l_{\mathcal{E}}(t_E) = l_{\mathcal{F}}(t_F)$ holds.

It is also obvious that π_E° can be broken down into the two parallel components E_l and E_p, and similarly π_F° can be decomposed into F_l and F_p.

In any HPB containing the tuple (π_E, π_F, f), any future behaviour of E_l has to be matched by behaviour of F_l, and vice versa. Similarly, any behaviour of E_p has to be matched by F_p, and also the other way around. Only then can it be ensured that the matching reflects the partial ordering correctly. But this amounts to the existence of two HPBs, one relating E_l and F_l, and the other relating E_p and F_p.

The second part of the lemma follows from a symmetrical argument. □

Lemma 3.4 (Completeness) *Let $\mathcal{E} = (\{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}, l_{\mathcal{E}})$ and $\mathcal{F} = (\{Y_j \stackrel{def}{=} F_j : j = 1, 2, \ldots, m\}, l_{\mathcal{F}})$ be two given BPP in ENF. If $\mathcal{E} \sim_{HPB} \mathcal{F}$ then there exists a successful tableau $T(X_1 = Y_1)$.*

Proof. Assume we have two BPP in ENF $\mathcal{E} = (\{X_i \stackrel{def}{=} E_i : i = 1, 2, \ldots, n\}, l_{\mathcal{E}})$ and $\mathcal{F} = (\{Y_j \stackrel{def}{=} F_j : j = 1, 2, \ldots, m\}, l_{\mathcal{F}})$ such that $\mathcal{E} \sim_{HPB} \mathcal{F}$. We shall show that there indeed exists a successful tableau $T(X_1 = Y_1)$.

The tableau rules are forward sound in the following sense. If we apply a rule to a pair of history-preserving bisimilar expressions, we can always find a rule instantiation such that the expressions related by the subgoals of the rule are history-preserving bisimilar as well. This is obvious for rule **Rec**, and immediately follows for rule **Match** from lemma 3.3.

Thus, starting from the root we can build a tableau such that every node relates two expressions that are history-preserving bisimilar. Since every tableau is finite, this construction will surely terminate. It is easy to see that two expressions that are related by unsuccessful terminal nodes cannot be history-preserving bisimilar,

so each terminal node will be successful. Hence, we have proved that there indeed exists a successful tableau. □

For the proof of soundness we give an alternative definition of HPB based on bisimulation approximations.

Definition 3.5 Let E, F be ENF expressions. A set $\mathcal{H}$ of triples (π_E, π_F, f) is a *history-preserving bisimulation approximation of degree* n for E, F if

(i) Whenever $(\pi_E, \pi_F, f) \in \mathcal{H}$, then π_E is a process of E, π_F is a process of F and f is a pomset isomorphism from $pom(\pi_E)$ onto $pom(\pi_F)$.

(ii) $(\pi_0(E), \pi_0(F), \emptyset) \in \mathcal{H}$.

(iii) Whenever $(\pi_E, \pi_F, f) \in \mathcal{H}$, $|E_{\pi_E}| < n$, and $\pi_E \xrightarrow{t_E} \pi'_E$ for some t_E, π'_E, then there exist t_F, π'_F, f' such that $\pi_F \xrightarrow{t_F} \pi'_F$, $(\pi'_E, \pi'_F, f') \in \mathcal{H}$ and $f' \restriction_{pom(\pi_E)} = f$.

(iv) Vice versa.

For two expressions E and F, we write $E \sim^n_{HPB} F$ iff there is a HPB approximation of degree n relating them.

With the standard argument we get the following characterization of HPB.

Lemma 3.6 *For image-finite systems we have*

$$\sim_{HPB} = \bigcap_{n=0}^{\infty} \sim^n_{HPB} .$$

Now we can state the essential lemma for soundness.

Lemma 3.7 (Backwards Soundness of Match) *Let E and F be two ENF process expressions. We have $E \sim^{n+1}_{HPB} F$, if the following holds,*

- *Whenever $E \xrightarrow{t_E} (E_l, E_p)$ then there exist $t_F, (F_l, F_p)$ such that $l_{\mathcal{E}}(t_E) = l_{\mathcal{F}}(t_F)$, $F \xrightarrow{t_F} (F_l, F_p)$, $E_l \sim^n_{HPB} F_l$ and $E_p \sim^n_{HPB} F_p$.*
- *Vice versa.*

Proof.

Imagine we are given label-preserving functions $f : T(E) \to T(F)$ and $g : T(F) \to T(E)$, and two families of HPB approximations of degree n, the family $\{\mathcal{H}_{(t_E,t_F)lR}\}_{(t_E,t_F)\in f\cup g}$ relating the pairs $localR(E, t_E)$ and $localR(F, t_F)$, and the family $\{\mathcal{H}_{(t_E,t_F)pR}\}_{(t_E,t_F)\in f\cup g}$ relating the pairs $parallelR(E, t_E)$ and $parallelR(F, t_F)$. The existence of these entities is guaranteed by the assumption of the lemma.

We shall now construct a HPB approximation of degree $n + 1$ for E and F based on these entities.

First we prefix all joint processes of the $\mathcal{H}_{(t_E,t_F)lR}$ with the tuple corresponding to the occurrence of the pair (t_E, t_F). This gives us a family of sets $\{\mathcal{H}'_{(t_E,t_F)lR}\}$, each defined by

$$\mathcal{H}'_{(t_E,t_F)lR} = \{pr^-((E, F), (t_E, t_F)) \circ \pi \mid \pi \in \mathcal{H}_{(t_E,t_F)lR}\}.$$

Now we define,

$$\mathcal{H} = \{pr((E,F),(\emptyset,\emptyset))\} \cup$$
$$\{\pi_l \mid\mid \pi_p \mid \pi_l \in \mathcal{H}'_{(t_E,t_F)lR} \text{ and } \pi_p \in \mathcal{H}_{(t_E,t_F)pR} \text{ for some } (t_E,t_F) \in f \cup g\}$$

It is easy to check that $\mathcal{H}$ is indeed a HPB approximation of degree $n+1$ relating E and F. □

Lemma 3.8 (Soundness) *Let $\mathcal{E} = (\{X_i \stackrel{def}{=} E_i : i = 1,2,\ldots,n\}, l_\mathcal{E})$ and $\mathcal{F} = (\{Y_j \stackrel{def}{=} F_j : j = 1,2,\ldots,m\}, l_\mathcal{F})$ be two given BPP in ENF. If there is a successful tableau $T(X_1 = Y_1)$ then $\mathcal{E} \sim_{HPB} \mathcal{F}$.*

Proof. Let us assume the contrary, i. e. that there is a successful tableau $T(X_1 = Y_1)$, but $X_1 \not\sim_{HPB} Y_1$. We shall show that this assumption leads to a contradiction.

If $X_1 \not\sim_{HPB} Y_1$ then by lemma 3.6 there is a least k such that $X_1 \sim^n_{HPB} Y_1$ for all $n < k$ and $X_1 \not\sim^n_{HPB} Y_1$ for all $n \geq k$.

Note that the tableau rules are backwards sound w. r. t. $\sim^n_{HPB}$. If we have a rule instantiation such that the related expressions of each subgoal are history-preserving bisimilar of approximation n, then the expressions related by the premise must be history-preserving bisimilar of approximation n, as well. This is obvious for the rule **Rec** and follows for **Match** from lemma 3.7. Lemma 3.7 actually gives us a strengthening: the ENF expressions related by the premise must be history-preserving bisimilar of approximation $n+1$.

Thus, in our assumed tableau we can trace a path π such that $\alpha \not\sim^k_{HPB} \beta$ for the related expressions α and β of each node. While tracing this path we can mark each node with the least l such that $\alpha \sim^n_{HPB} \beta$ for all $n < l$ and $\alpha \not\sim^n_{HPB} \beta$ for all $n \geq l$. Note that the sequence of these measures along π is strictly decreasing due to instantiations of **Match**.

Now consider the terminal node n_t of π. Since the tableau is successful it must be a successful terminal, i. e. it is labelled by "$\mathbf{0} = \mathbf{0}$", or by "$X = Y$" and we have an ancestor node n_a labelled by "$X = Y$" as well. The first case cannot be possible since clearly $\mathbf{0} \sim_{HPB} \mathbf{0}$. So let us consider the second case. Let k_{n_t} be the measure of n_t, and k_{n_a} the measure of n_a respectively. Observe that there must be an instantiation of **Match** between n_a and n_t on our path π, and hence we have $k_{n_t} < k_{n_a}$. But this is clearly a contradiction.

□

With the above results the decidability of HPB is straightforward. By soundness and completeness we only have to check whether there exists a successful tableau. Since there is only a finite number of tableaux for any two given BPPs we can simply do this by exhaustive search.

Theorem 3.9 *History-preserving bisimilarity is decidable for BPP.*

4 Coincidence of History-Preserving Bisimilarity and Distributed Bisimilarity for BPP

Distributed bisimulation [5,6] is the natural notion of bisimulation corresponding to Castellani's distributed transition semantics. It refines classical bisimulation by requiring that local residuals and concurrent residuals are related separately. The definition is based on the distributed transition relation, which is given via operational semantics. We shall not give the corresponding SOS rules here but we derive distributed transitions from our occurrence net semantics, i. e. we make use of our notation $E \stackrel{t_E}{\rightarrow} (E_l, E_p)$. Note that it would be a straightforward task to show that the transition relation thus defined indeed coincides with the distributed transition relation of [5].

Definition 4.1 A relation $\mathcal{D}$ over ENF process expressions is a *distributed bisimulation* if for any $(E, F) \in \mathcal{D}$ we have

(i) Whenever $E \stackrel{t_E}{\rightarrow} (E_l, E_p)$ for some t_E, (E_l, E_p), then there exist t_F, (F_l, F_p) such that $l_{\mathcal{E}}(t_E) = l_{\mathcal{F}}(t_F)$, $F \stackrel{t_F}{\rightarrow} (F_l, F_p)$, $(E_l, F_l) \in \mathcal{D}$, and $(E_p, F_p) \in \mathcal{D}$.

(ii) Vice versa.

We say two ENF expressions are distributed bisimilar iff there is a distributed bisimulation relating them.

It follows directly from the definition that the tableau rules for HPB are forward and backwards sound for distributed bisimulation. Hence, the tableau provides a decision procedure for distributed bisimilarity just as well, which immediately establishes the coincidence of the two notions for BPP.

Theorem 4.2 *History-preserving bisimilarity and distributed bisimilarity coincide for BPP.*

So it is not surprising that the tableau is very similar to the one employed in Christensen's proof of the decidability of distributed bisimilarity. The major new ingredient in the proof for HPB are the lemmas for forward and backwards soundness of **Match**. They prove that for BPP the distributed and the partial order view are equivalent.

Our technical framework is slightly different. This comes from the fact that Christensen makes use of his BPP standard normal form, where every defining expression is of the form $\sum_{j=1}^{n_i} a_{ij}.\alpha_{ij} \lfloor \beta_{ij}$ such that each α_{ij}, β_{ij} is a parallel merge of variables. The left merge operator $\lfloor$ acts like the usual parallel merge under the constraint that the first action must come from the left process. So, due to the use of this normal form the local and concurrent residuals are already separated out in the process expressions of Christensen's tableau rules.

5 Decidablity of Hereditary History-Preserving Bisimilarity for BPP

As in section 3 we start with the introduction of a notion that is crucial for our proof. From an ENF expression E we can easily read which maximal steps of parallel actions one can take from E. This concept is formally captured in the following definition.

Definition 5.1 Let E be an ENF process expression. An *independence clique* of E is a set $cl_E = \{t_1, t_2, \ldots, t_n\} \subseteq T(E)$ such that for some E', $E \stackrel{w}{\rightarrow} E'$, where $w = t_1 \ldots t_n$.

A set $mcl_E \subseteq T(E)$ is a *maximal independence clique* of E iff mcl_E is an independence clique, and mcl_E is maximal, in the sense that adding any other transition would violate the first condition.

We denote the set of independence cliques of E by $Cliques(E)$, and the set of maximal independence cliques of E by $MCliques(E)$.

In the following we will use the word clique as an abbreviation for independence clique.

Again, we employ the tableau technique to prove our result. The tableau rules can be found in figure 3. The rule **Match** is now more complicated. It reflects the idea behind the proof: we match all the possible maximal cliques of two related ENF expressions E and F in one step. This makes it possible to impose conditions on the matching of the transitions of E and F that ensure that the backtracking requirement is satisfied within the range of this match. If we additionally make sure that whenever the same pair of transitions is matched we use the same matching for the corresponding local components, then the backtracking condition will be globally satisfied.

Our terminal conditions are very similar to the ones of the tableau for HPB. A node $n : label$ is a *successful terminal* if one of the following conditions holds:

(i) $label =$ "$\mathbf{0} = \mathbf{0}$".

(ii) $label =$ "$X = Y$", and there is an ancestor node n_a above n in the tableau such that n_a is labelled with "$X = Y$" as well.

A node $n : label$ is an *unsuccessful terminal* if the following condition holds:

(i) $label =$ "$E = F$", where E and F are ENF expressions, and a family of bijections $\mathcal{B}$ as required by rule **Match** does not exist.

Now we establish forward and backwards soundness of the rule **Match**. For the proofs of finiteness, completeness and soundness we can then use exactly the same arguments as in the corresponding proofs of the previous section.

Lemma 5.2 (Forward Soundness of Match) *Let E and F be two ENF process expressions such that $E \sim_{HHPB} F$. Then there is a family of bijections $\mathcal{B} = \{f_i\}$ satisfying the conditions of figure 4, and whenever $(t_E, t_F) \in \bigcup f_i$ we have $localR(E, t_E) \sim_{HHPB} localR(F, t_F)$.*

$$\textbf{Rec} \quad \frac{X_i = Y_j}{E_i = F_j}$$

$$\textbf{Match} \quad \frac{E = F}{\{localR(E, t_E) = localR(F, t_F)\}_{(t_E, t_F) \in \bigcup f_i}}$$

where $\mathcal{B} = \{f_i\}$ is a family of bijections
satisfying the conditions of figure 4.

Fig. 3. Tableau Rules for HHPB

A family of bijections $\mathcal{B} = \{f_i\}$ satisfying the following conditions:

(i) For each f_i, we have $Domain(f_i) \in MCliques(E)$ and $Range(f_i) \in MCliques(F)$.

(ii) Each f_i is label-preserving, i. e. for each $(t_E, t_F) \in f_i$ we have $l_{\mathcal{E}}(t_E) = l_{\mathcal{F}}(t_F)$.

(iii) The family of f_i gives a match for all possible cliques of E, and F respectively, i. e.
  (a) Forall $mcl_E \in MCliques(E)$ there exists f_i with $Domain(f_i) = mcl_E$.
  (b) Forall $mcl_F \in MCliques(F)$ there exists f_i with $Range(f_i) = mcl_F$.

(iv) Forall $\sigma \in \mathcal{P}(T(E) \times T(F))$ such that $\sigma \subseteq f_i$ for some f_i, we have
  (a) Whenever $pr(E, \sigma_E) \stackrel{t_E}{\rightarrow} \pi_E$ for some t_E, π_E, then there exist t_F, f_j with $\sigma \cup (t_E, t_F) \subseteq f_j$.
  (b) Vice versa.

Fig. 4. Conditions for the family of bijections $\mathcal{B}$

Proof. Assume $\mathcal{H}$ to be a HHPB relating E and F. Let us first show that a family of bijections $\mathcal{B}$ exists as required. As our family of bijections we take
$\mathcal{B} = \{f \mid \exists(\pi_E, \pi_F, f) \in \mathcal{H}$ s. t. E_{π_E} corresponds to some $mcl_E \in MCliques(E)\}$.

Since each $f_i \in \mathcal{B}$ is a pomset isomorphism this clearly defines a family of label-preserving bijections. Now let us check the remaining conditions (1), (3) and (4). The first part of condition (1) follows directly from the definition of $\mathcal{B}$. To see that the second part also holds, i. e. that for each f_i, $Range(f_i) \in MCliques(F)$, first note that each f_i can reflect the partial order of its domain only correctly if its range is a clique of F. Secondly, observe that this clique must be maximal. If there was a remaining transition t_F such that $Range(f_i) \cup \{t_F\} \in Cliques(F)$, there could be no match of t_F at the tuple (π_E, π_F, f_i) corresponding to f_i in our HPB $\mathcal{H}$, since $Domain(f_i)$ is by definition a maximal clique.

Condition (3a) follows immediately when considering that $\mathcal{H}$ is a HPB, and therefore must contain matches for all possible behaviour of E. Surely it contains matches for the maximal cliques of E.

For condition (3b) consider that in any HPB each maximal clique of F has to

be matched to a maximal clique of E (by an argument similar to the one used for condition (1)). Of course, $\mathcal{H}$ contains matches for the maximal cliques of F, and together with the previous observation this implies (3b).

To check condition (4a) imagine we have $\sigma \in \mathcal{P}(T(E) \times T(F))$ such that $\sigma \subseteq f_i$ for some f_i. Due to the way the family of f_i is defined we have a corresponding tuple $(\pi_E, \pi_F, f_i) \in \mathcal{H}$. But by proposition (v) of the definition of HHPB this means we also have $(pr(E, \sigma_E), pr(F, \sigma_F), \sigma) \in \mathcal{H}$. Now we can apply proposition (iii) of the definition of HHPB: whenever we have $pr(E, \sigma_E) \xrightarrow{t_E} \pi'_E$ for some π'_E, there is a match t_F, such that $pr(F, \sigma_F) \xrightarrow{t_F} \pi'_F$ for some π'_F and $(\pi'_E, \pi'_F, \sigma \cup (t_E, t_F)) \in \mathcal{H}$. Certainly from this new tuple we have further matches in $\mathcal{H}$ up to the next maximal clique. So, indeed we have $\sigma \cup (t_E, t_F) \subseteq f_j$ for some t_F, f_j.

Condition (4b) can be proved by a symmetrical argument.

It remains to show that $localR(E, t_E) \sim_{HHPB} localR(F, t_F)$ for every pair $(t_E, t_F) \in \bigcup f_i$. Whenever t_E and t_F are matched against each other in a HHPB, the remaining behaviour of E that is dependent on t_E must be matched by remaining behaviour of F that is dependent on t_F, and vice versa. But this amounts to the existence of a HHPB for $localR(E, t_E)$ and $localR(F, t_F)$. □

Before we establish the essential lemma for soundness we need to define bisimulation approximations for HHPB.

Definition 5.3 A *hereditary history-preserving bisimulation approximation of degree* n is a history-preserving bisimulation approximation of degree n that further satisfies

(v) Whenever $\pi \in \mathcal{H}$ and $\pi' \subseteq \pi$ for some π', then $\pi' \in \mathcal{H}$.

For two ENF expressions E and F, we write $E \sim^n_{HHPB} F$ iff there is a HHPB approximation of degree n relating them.

Again, we have for image-finite systems:

$$\sim_{HHPB} = \bigcap_{n=0}^{\infty} \sim^n_{HHPB} .$$

Lemma 5.4 (Backwards Soundness of Match) *Let E and F be two ENF process expressions. Then we have $E \sim^{n+1}_{HHPB} F$, if there is a family of bijections $\mathcal{B} = \{f_i\}$ satisfying the conditions of figure 4, and whenever $(t_E, t_F) \in \bigcup f_i$ we have $localR(E, t_E) \sim^n_{HHPB} localR(F, t_F)$.*

Proof. Imagine we are given a family of bijections $\mathcal{B} = \{f_i\}$ as required, and a family of HHPB approximations of degree n, $\{\mathcal{H}_{(t_E,t_F)}\}_{(t_E,t_F)\in\bigcup f_i}$, where each $\mathcal{H}_{(t_E,t_F)}$ relates the processes $localR(E, t_E)$ and $localR(F, t_F)$. We shall construct a HHPB approximation of degree $n + 1$ for E and F, based on this family of n approximations.

First we attach the tuples of each $\mathcal{H}_{(t_E,t_F)}$ to the process corresponding to the occurrence of (t_E, t_F). This results in a family of sets $\{\mathcal{H}'_{(t_E,t_F)}\}$, each defined by

$$\mathcal{H}'_{(t_E,t_F)} = \{pr^-((E, F), (t_E, t_F)) \circ \pi \mid \pi \in \mathcal{H}_{(t_E,t_F)}\}.$$

Now we define,

$$\mathcal{H} = \{pr(parallelR((E,F),\sigma),(\emptyset,\emptyset)) \mid\mid \pi_{t_1} \mid\mid \pi_{t_2} \mid\mid \ldots \mid\mid \pi_{t_n} \mid \sigma = \{t_1, t_2, \ldots, t_n\} \subseteq f_i \text{ for some } f_i, \text{ and } \pi_{t_k} \in \mathcal{H}'_{t_k} \text{ for each } k \in \{1, \ldots, n\}\},$$

where $parallelR$ is the function of the previous section generalized to cliques of transitions (and used analogously to the conventions described in section 2). It is straightforward to check that $\mathcal{H}$ is indeed a HHPB approximation of degree $n+1$.□

We can now proceed analogously to the proof of the decidability of HPB. Finiteness of our tableau system can be established following the proof of lemma 3.2. With the two lemmas 5.2 and 5.4, completeness and soundness can be proved by using the arguments of the lemmas 3.4 and 3.8. Again, the decidability of HHPB follows from these three properties of our tableau system.

Theorem 5.5 *Hereditary history-preserving bisimilarity is decidable for BPP.*

6 Final Remarks

First of all, the concept of ENF needs further development. We have already seen a net representation of BPPs in ENF using possibly infinitary 1-safe Petri nets. It would be nice to have a finite net representation based on P/T nets, such that the unfolding of a P/T net characterization would coincide with the unfolding of the BPP it represents. There is an obvious way of how a BPP in ENF can be translated into a P/T net. However, the unfolding of the P/T net does not coincide with the unfolding of the BPP (although, the two unfoldings are probably equivalent under HHPB).

Regarding the proofs of the decidability of plain and hereditary HPB there are mainly two points left for further investigation. One is to establish the complexity of our decision procedures. The other point is to compare the proof of plain HPB given in this paper to the proof of the decidability of causal bisimulation in [22].

Finally, let me remark that HPB and HHPB coincide for BPP in full standard form [14]. However, this is not the case for the entire class of BPP, as it is shown by example 1.7 of [3]. [4]

Acknowledgement

I would like to thank the many people with whom I have discussed the notions of plain and hereditary HPB, in particular I thank Colin Stirling for his advice on my BPP work, Astrid Kiehn for providing me with her notes on distributed bisimilarity, and Ilaria Castellani and Rob van Glabbeek for their helpful comments at EXPRESS. I also thank two of the anonymous referees for valuable comments

[4] Many thanks to Rob van Glabbeek for pointing this out to me.

which drew my attention to the coincidence of distributed and history-preserving bisimilarity.

References

[1] Aceto, L., *History preserving, causal and mixed-ordering equivalence over stable event structures*, Fund. Inform. **17** (1992), pp. 319–331.

[2] Aceto, L., *Relating distributed, temporal and causal observations of simple processes*, Fundamenta Informaticae **17** (1992), pp. 369–397.

[3] Bednarczyk, M., *Hereditary history preserving bisimulation or what is the power of the future perfect in program logics*, Technical report, Polish Academy of Sciences, Gdansk (1991).

[4] Best, E., R. Devillers, A. Kiehn and L. Pomello, *Concurrent bisimulations in Petri nets*, Acta Inform. **28** (1991), pp. 231–264.

[5] Castellani, I., "Bisimulations for Concurrency," Ph.D. thesis, University of Edinburgh (1988).

[6] Castellani, I. and M. Hennessy, *Distributed bisimulations*, Journal of the ACM **36** (1989), pp. 887–911.

[7] Christensen, S., *Distributed bisimularity is decidable for a class of infinite state-space systems*in: *CONCUR '92 (Stony Brook, NY, 1992)*, Springer, Berlin, 1992 pp. 148–161.

[8] Christensen, S., "Decidability and Decomposition in Process Algebras," Ph.D. thesis, University of Edinburgh (1993).

[9] Christensen, S., Y. Hirshfeld and F. Moller, *Bisimulation equivalence is decidable for basic parallel processes*in: *CONCUR '93 (Hildesheim, 1993)*, Springer, Berlin, 1993 pp. 143–157.

[10] Degano, P., R. De Nicola and U. Montanari, *Partial orderings descriptions and observations of nondeterministic concurrent processes*, in: *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, LNCS **354**, 1989, pp. 438–466.

[11] Engelfriet, J., *Branching processes of Petri nets*, Acta Inform. **28** (1991), pp. 575–591.

[12] Esparza, J., *Petri nets, commutative context-free grammars, and basic parallel processes*, Fund. Inform. **31** (1997), pp. 13–25.

[13] Esparza, J. and A. Kiehn, *On the model checking problem for branching time logics and basic parallel processes*in: *Proceedings of Computer Aided Verification (CAV '95), 7th International Conference*, Springer, Berlin, 1995 pp. 353–366.

[14] Fröschle, S., *On the decidability problem of strong history-preserving bisimulation* (1998), progress Report.

[15] Fröschle, S. and T. Hildebrandt, *On plain and hereditary history-preserving bisimulation* (1999), to appear at MFCS'99.

[16] Gorrieri, R. and U. Montanari, *SCONE: a simple calculus of nets*in: *CONCUR' 90 (Amsterdam, 1990)*, Springer, Berlin, 1990 pp. 2–30.

[17] Jategaonkar, L. and A. R. Meyer, *Deciding true concurrency equivalences on safe, finite nets*, Theoret. Comput. Sci. **154** (1996), pp. 107–143.

[18] Joyal, A., M. Nielsen and G. Winskel, *Bisimulation from open maps*, Inform. and Comput. **127** (1996), pp. 164–185.

[19] Jurdziński, M. and M. Nielsen, *Hereditary history preserving bisimilarity is undecidable*, Research Series RS-99-19, BRICS, Department of Computer Science, University of Aarhus (1999), 18 pp.

[20] Jurdziński, M. and M. Nielsen, *Hereditary history preserving simulation is undecidable*, Research Series RS-99-1, BRICS, Department of Computer Science, University of Aarhus (1999), 15 pp.

[21] Kiehn, A., *A note on distributed bisimulations* (1999).

[22] Kiehn, A. and M. Hennessy, *On the decidability of non-interleaving process equivalences*in: *CONCUR '94: concurrency theory (Uppsala, 1994)*, Springer, Berlin, 1994 pp. 18–33.

[23] Moller, F., *Infinite results*in: *CONCUR '96: concurrency theory (Pisa)*, Springer, Berlin, 1996 pp. 195–216.

[24] Montanari, U. and M. Pistore, *Minimal transition systems for history-preserving bisimulation*in: *STACS 97 (Lübeck)*, Springer, Berlin, 1997 pp. 413–425.

[25] Nielsen, M., G. Plotkin and G. Winskel, *Petri nets, event structures and domains. I*, Theoret. Comput. Sci. **13** (1981), pp. 85–108.

[26] Rabinovich, A. and B. A. Trakhtenbrot, *Behavior structures and nets*, Fund. Inform. **11** (1988), pp. 357–403, concurrency.

[27] Sunesen, K. and M. Nielsen, *Behavioural equivalence for infinite systems—partially decidable!*in: *Application and theory of Petri nets 1996 (Osaka, 1996)*, Springer, Berlin, 1996 pp. 460–479.

[28] van Glabbeek, R. and U. Goltz, *Equivalence notions for concurrent systems and refinement of actions (extended abstract)*in: *Mathematical foundations of computer science 1989 (Porpolhk abka-Kozubnik, 1989)*, Lecture Notes in Comput. Sci. **379**, Springer, Berlin, 1989 pp. 237–248.

[29] Vogler, W., *Deciding history preserving bisimilarity*in: *Automata, languages and programming (Madrid, 1991)*, Lecture Notes in Comput. Sci. **510**, Springer, Berlin, 1991 pp. 495–505.

[30] Vogler, W., *Generalized OM-bisimulation*, Inform. and Comput. **118** (1995), pp. 38–47.