

Composition and Decomposition in True-Concurrency

Sibylle Fröschle*

Institute of Informatics, University of Warsaw, Poland
sib@mimuw.edu.pl

Abstract. The idea of composition and decomposition to obtain computability results is particularly relevant for true-concurrency. In contrast to the interleaving world, where composition and decomposition must be considered with respect to a process algebra operator, e.g. parallel composition, we can directly recognize whether a truly-concurrent model such as a labelled asynchronous transition system or a 1-safe Petri net can be dissected into independent ‘chunks of behaviour’. In this paper we introduce the corresponding concept ‘decomposition into independent components’, and investigate how it translates into truly-concurrent bisimulation equivalences. We prove that, under a natural restriction, history preserving (hp), hereditary hp (hhp), and coherent hhp (chhp) bisimilarity are decomposable with respect to prime decompositions. Apart from giving a general proof technique our decomposition theory leads to several coincidence results. In particular, we resolve that hp, hhp, and chhp bisimilarity coincide for ‘normal form’ basic parallel processes.

1 Introduction

In the finite-state world truly-concurrent problems are typically harder than their interleaving counterparts. This is demonstrated by the following examples. Model-checking CTL is well-known to be polynomial-time but model-checking CTL_P is NP-hard [1]. The problem of synthesizing controllers for discrete event systems is decidable in an interleaving setting and can be computed in polynomial-time; in a truly-concurrent setting the problem is undecidable [2]. Classical bisimilarity is polynomial-time decidable while *hereditary history preserving (hhp) bisimilarity* has been proved undecidable [3]; plain *history preserving (hp) bisimilarity* is decidable [4] but has been shown DEXPTIME-complete [5, 6].

There is, however, a positive trend for true-concurrency in the *infinite*-state world. The above effect seems reversed for *basic parallel processes (BPP)*. Under interleaving semantics a small fragment of a logic equivalent to CTL^* is undecidable for *very basic BPP*; under partial order interpretation the full logic is decidable for BPP [7]. Trace equivalence on BPP is undecidable but pomset trace and location trace equivalence on BPP are shown decidable in [8]. Classical bisimilarity on BPP is PSPACE-complete [9, 10]; in contrast, for BPP, distributed bisimilarity, and with it hp bisimilarity, are polynomial-time decidable [11]. The positive trend is further confirmed by results of

* This work was supported by the EPSRC Grants GR/M84763 and GR/R16891, and the European Community Research Training Network ‘GAMES’.

[12, 13]: hhp bisimilarity on BPP is decidable and coincides with its strengthening to *coherent hhp bisimilarity*.

We can explain this discrepancy as follows. Models such as *labelled asynchronous transition systems (lats)* [14] or *labelled 1-safe Petri nets (net systems)* faithfully capture how the transitions of a system are related concerning concurrency and conflict. The way we allow concurrency and conflict to interact will directly impact on the computational power of truly-concurrent equivalences and logics. The negative results of [2] and [3] build on the insight that truly-concurrent models have the power to encode tiling systems. If the interplay between concurrency and conflict is restricted this power can be lost [15], and a truly-concurrent concept may be particularly natural to decide. BPP are infinite-state but, under truly-concurrent semantics, they have a simple tree-like structure, which has turned out to be directly exploitable: e.g. the decidability results of [8] follow by a reduction to the equivalence problem of recognizable tree languages.

In this paper we advocate the following thesis. System classes with a restricted interplay between concurrency and conflict often have characteristic decomposition properties. These might translate into truly-concurrent equivalences or logics in a very concrete way, and thereby allow us to decide the respective concept by a ‘divide and conquer’ approach.

The idea of decomposition provides one of the crucial techniques to establish decidability and upper complexity bounds in infinite-state verification. For example, the polynomial-time decision procedure for classical bisimilarity on normed BPP [16] is based on the following insight:¹ any normed BPP can be expressed uniquely, up to bisimilarity, as a parallel composition of prime factors [18]. A process is *prime* if it is not the nil process and it is irreducible with respect to parallel composition, up to bisimilarity. Such a decomposition theory translates into cancellation properties of the form “ $P \parallel Q \sim R \parallel Q$ implies $P \sim R$ ”, which provide the means to reduce pairs of processes to compare into smaller pairs of processes to check. Questions about prime decomposability were first addressed by Milner and Moller in [19].

In the interleaving world, decomposition must be considered with respect to a process algebra operator, e.g. parallel composition, and the behavioural equivalence of choice: can a process term P be expressed as a process term Q of particular form, a parallel composition of prime processes, such that P and Q are bisimilar? In contrast, in true-concurrency, decomposition can be considered at the level of the semantic model: we can directly recognize whether a lats or net system can be dissected into independent ‘chunks of behaviour’. Having fixed a specific decomposition view on the level of the model we can then separately investigate whether this view translates into a given equivalence. For example, we might suspect: if two parallel compositions of sequential systems, say S and S' , are equivalent under a truly-concurrent bisimilarity then there is a one-to-one correspondence between the components of S and those of S' such that related components are equivalent. For classical bisimilarity this decomposition property will certainly not hold: $a \parallel b$ is bisimilar to $a.b + b.a$.

There are two axioms of independence: (1) If two independent transitions can occur consecutively then they can also occur in the opposite order. (2) If two independent

¹ Very recently this result has been improved to $O(n^3)$ by an algorithm that does not use decomposition in this sense [17].

transitions are enabled at the same state then they can also occur one after the other. This indicates that decomposition is inherently connected to the shuffling of transitions: the behaviour of a system corresponds to the shuffle product of the behaviour of its independent components. Therefore, decomposition theorems provide an important tool to establish coincidence between hp, hhp, and chhp bisimilarity: proving that the three equivalences coincide amounts to proving that whenever two systems are hp bisimilar there exists a hp bisimulation that satisfies specific shuffle properties, the hereditary and coherent condition.

The contribution of this paper is threefold: (1) We transfer the idea of prime decomposition to the truly-concurrent world. (2) We analyse whether this concept translates into truly-concurrent bisimulation equivalences. We show that, under a natural restriction, hp, and also, hhp and chhp bisimilarity are indeed decomposable with respect to prime decompositions. (3) We apply our decomposition theory to obtain coincidence results. In particular, this gives us several positive results for hhp bisimilarity, a concept which is renowned for being difficult to analyse. In more detail, after presenting the necessary definitions in Section 2, we proceed as follows.

In Section 3 we introduce the notion ‘*decomposition into independent components*’ and a corresponding concept of *prime component* for the model of lats’; components are defined as concrete sub-systems of the respective lats. We show that every non-empty system uniquely decomposes into its set of prime components.

In Section 4 we show that hp, hhp, and chhp bisimilarity are *composable* with respect to decompositions in the following sense: assume two systems S_1, S_2 , each decomposed into a set of independent components; whenever we can exhibit a one-to-one correspondence between the components of S_1 and those of S_2 such that related components are hp (hhp, chhp) bisimilar then S_1 and S_2 are hp (hhp, chhp) bisimilar. This is straightforward but guarantees the soundness of our decomposition approach. It is related to congruence in the process algebra world: if $P \sim P'$ and $Q \sim Q'$ then $P \parallel Q \sim P' \parallel Q'$.

Section 5 is the core of the paper: we analyse whether hp, hhp, and chhp bisimilarity are *decomposable* in the converse sense. We demonstrate that hp bisimilarity is *not* decomposable with respect to prime decompositions. However, we identify a natural restriction under which this is indeed given for hp, and also, hhp and chhp bisimilarity: for systems whose prime components are, what we shall call, *concurrent step connected (csc)*. We obtain: whenever two *csc-decomposable systems* S_1, S_2 are hp (hhp, chhp) bisimilar then there is a one-to-one correspondence between the prime components of S_1 and those of S_2 such that related components are hp (hhp, chhp) bisimilar. The proof of this statement is non-trivial. In particular, we require the combinatorial argument of Hall’s Marriage Theorem.

In Section 6 we apply our (de)composition theory to prove several coincidence results. As an immediate consequence we obtain coincidence between hp, hhp, and chhp bisimilarity for *parallel compositions of sequential systems*. Most interesting is, perhaps, that this intuitive result has turned out non-trivial to prove, and that the key insight behind it is of general significance. By employing our (de)composition theory in an inductive argument we extend the coincidence result to the class *concurrency-degree bounded communication-free net systems*.

Most importantly, we resolve that hp, hhp, and chhp bisimilarity coincide for the *simple basic parallel processes (SBPP)* of [7]. SBPP correspond to BPP in normal form, which in the interleaving world represent the entire BPP class; in true-concurrency they form a strictly smaller class. The coincidence for SBPP complements the positive results already achieved for (h)hp bisimilarity on BPP. Via [11] it follows that hhp bisimilarity on SBPP is polynomial-time decidable. Since hp and hhp bisimilarity do *not* coincide for BPP in general, the coincidence for SBPP underlines that SBPP and BPP do behave differently in the truly-concurrent world.

In Section 7 we conclude the paper and point to future research. Most of the proofs are kept informal in this extended abstract; a detailed account can be found in the appendix. Our primary model is *lats*², but we also informally employ net systems, which can be understood as a class of *lats*²; their definition is provided in Appendix A.1.

2 Preliminaries

Systems. A *labelled (coherent) asynchronous transition system* (for this paper simply *system*) is defined as a structure $S = (S_S, s_S^i, T_S, \rightarrow_S, I_S, l_S)$, where S_S is a set of *states* with *initial state* $s_S^i \in S_S$, T_S is the set of *transitions*², $\rightarrow_S \subseteq S_S \times T_S \times S_S$ is the *transition relation*, $I_S \subseteq T_S \times T_S$, the *independence relation*, is an irreflexive, symmetric relation, and $l_S : T_S \rightarrow Act$ is the *labelling function*, where $Act = \{a, b, \dots\}$ is a set of *actions*, such that

1. $t \in T_S \implies \exists s, s' \in S_S. s \xrightarrow{t}_S s'$,
2. $s \xrightarrow{t}_S s' \ \& \ s \xrightarrow{t}_S s'' \implies s' = s''$,
3. $t_1 I_S t_2 \ \& \ s \xrightarrow{t_1}_S s_1 \ \& \ s_1 \xrightarrow{t_2}_S u \implies \exists s_2. s \xrightarrow{t_2}_S s_2 \ \& \ s_2 \xrightarrow{t_1}_S u$, and
4. $t_1 I_S t_2 \ \& \ s \xrightarrow{t_1}_S s_1 \ \& \ s \xrightarrow{t_2}_S s_2 \implies \exists u. s_1 \xrightarrow{t_2}_S u \ \& \ s_2 \xrightarrow{t_1}_S u$.

We lift \rightarrow_S to sequences of transitions in the usual way. We also lift I_S to sequences and sets of transitions, e.g. we write $t_1 \dots t_n I_S t'_1 \dots t'_m$ iff $t_i I_S t'_j$ for all $i \in [1, n]$, $j \in [1, m]$. In this paper we assume a further axiom:

5. $s \in S_S \implies \exists w \in T_S^*. s_S^i \xrightarrow{w} s$.

Axiom (1) says that every transition can occur from some state, and axiom (2) that the occurrence of a transition at a state leads to a unique state. Axioms (3) and (4) express the two axioms of independence mentioned in the introduction. Our additional axiom (5) specifies that every state is reachable from the initial state. A system S is *finite* iff S_S and T_S are finite sets. S is *empty* iff $T_S = \emptyset$, and *non-empty* otherwise.

Let S be a system, and $s \in S_S$. The transitions of $T_c \subseteq T_S$ are *concurrently enabled* at s , $T_c \in cenabl_S(s)$, iff $\forall t \in T_c. \exists s'. s \xrightarrow{t} s'$ and $\forall t, t' \in T_c. t \neq t' \implies t I_S t'$. We define the *smallest upper bound on the number of transitions that are concurrently enabled* at s by $cbound_S(s) = \min\{\kappa \mid \forall T_c \in cenabl_S(s). |T_c| \leq \kappa\}$. S is *concurrency-degree finite* iff for each $s \in S_S$, $cbound_S(s) \in \mathbf{N}_0$. E.g., finitely branching systems are always concurrency-degree finite. **We only consider systems that are concurrency-degree finite.**

² in the sense of Petri net boxes

Partial Order Runs. A *pomset* is a labelled partial order; specified via a labelled strict order, it is a tuple $p = (E_p, <_p, l_p)$, where E_p is a set of *events*, $<_p$ a strict order relation on E_p , and l_p a labelling function $l_p : E_p \rightarrow Act$. A function g is an *isomorphism* between pomset p and pomset q iff $g : E_p \rightarrow E_q$ is a bijection such that (1) $l_p = l_q \circ g$, and (2) $e <_p e'$ iff $g(e) <_q g(e')$ for all $e, e' \in E_p$.

Assume a system S . Let $r = t_1 t_2 \dots t_n \in T_S^*$ be a sequence of transitions. We write $|r|$ for the length of r , that is $|r| = n$; for any $i \in [1, |r|]$ we denote the i th transition of r , t_i , by $r[i]$. r is a *run* of S , $r \in Runs(S)$, iff $s_S^i \xrightarrow{r} s$ for some state $s \in S_S$. The *pomset* of r , $pom(r)$, has as events the integers from 1 to n , where the label of event i is $l_S(t_i)$, and the strict ordering is the transitive closure of the following ‘‘proximate cause’’ relation: event i *proximately causes* event j , written $i <_r^{prox} j$, iff $i < j$ and t_i and t_j are *not independent* in S . We denote this strict ordering on $[1, n]$ by $<_r$.

Hp, Hhp, and Chhp Bisimilarity. Hp bisimilarity relates two systems whose behaviour can be bisimulated while preserving the labelling of transitions and the causal dependencies between them. Technically, this can be realized by basing hp bisimulation on pairs of *synchronous runs* [5]: intuitively, two runs are *synchronous* if their induced pomsets are isomorphic, and both runs correspond to the same linearization of the associated pomset isomorphism class. Formally, this amounts to: let S_1, S_2 be two systems; $r_1 \in Runs(S_1)$ and $r_2 \in Runs(S_2)$ are *synchronous*, $(r_1, r_2) \in SRuns(S_1, S_2)$, iff the identity function on $[1, |r_1|]$ is an isomorphism between $pom(r_1)$ and $pom(r_2)$. A set $\mathcal{H} \subseteq SRuns(S_1, S_2)$ is *prefix-closed* iff $(r_1 t_1, r_2 t_2) \in \mathcal{H}$ implies $(r_1, r_2) \in \mathcal{H}$. As noted in [20] it is safe to restrict our attention to prefix-closed hp bisimulations.

Hhp bisimilarity is obtained from hp bisimilarity by the addition of a *backtracking* requirement, and chhp bisimilarity furthermore imposes a *padding* requirement. These conditions reflect the first and, respectively, second axiom of independence.

Definition 1. Let S_1 and S_2 be two systems. A history preserving (hp) bisimulation relating S_1 and S_2 is a prefix-closed relation $\mathcal{H} \subseteq SRuns(S_1, S_2)$ that satisfies:

1. $(\varepsilon, \varepsilon) \in \mathcal{H}$.
2. If $(r_1, r_2) \in \mathcal{H}$ and $r_1 t_1 \in Runs(S_1)$ for some $t_1 \in T_1$, then there is $t_2 \in T_2$ such that $(r_1 t_1, r_2 t_2) \in \mathcal{H}$.
3. Vice versa.

A hp bisimulation \mathcal{H} is *hereditary (h)* when it further satisfies:

4. If $(r_1 t_1 w_1, r_2 t_2 w_2) \in \mathcal{H}$ for some $w_1 \in T_1^*$, $w_2 \in T_2^*$, $t_1 \in T_1$, and $t_2 \in T_2$ such that $|w_1| = |w_2|$, $t_1 I_1 w_1$ (or $t_2 I_2 w_2$ equivalently), then $(r_1 w_1, r_2 w_2) \in \mathcal{H}$.

A hhp bisimulation \mathcal{H} is *coherent (c)* when it further satisfies:

5. If $(r_1 w_1, r_2 w_2), (r_1 t_1, r_2 t_2) \in \mathcal{H}$ for some $w_1 \in T_1^*$, $w_2 \in T_2^*$, $t_1 \in T_1$, and $t_2 \in T_2$ such that $|w_1| = |w_2|$, $t_1 I_1 w_1$, and $t_2 I_2 w_2$, then $(r_1 t_1 w_1, r_2 t_2 w_2) \in \mathcal{H}$.

S_1 and S_2 are ((c)h)hp bisimilar, written $S_1 \sim_{((c)h)hp} S_2$, iff there exists a ((c)h)hp bisimulation relating them. Given two systems S_1 and S_2 , we also use $\sim_{((c)h)hp}$ to denote the set $\bigcup \{ \mathcal{H} : \mathcal{H} \text{ is a } ((c)h)hp \text{ bisimulation relating } S_1 \text{ and } S_2 \}$. (Note: chhp bisimulations are not closed under union; so, \sim_{chhp} is not necessarily the largest chhp bisimulation.)

Further Concepts. Let A, B be alphabets. For $r \in A^*$, if $B \subseteq A$, let $r \uparrow B$ denote the sequence obtained by erasing from r all occurrences of letters which are not in B . If $B = T_c$ for some system c we write $r \uparrow c$ short for $r \uparrow T_c$.

The *shuffle* of n words $u_1, \dots, u_n \in A^*$ is the set $u_1 \otimes \dots \otimes u_n$ of all words of the form $u_{1,1}u_{2,1} \dots u_{n,1}u_{1,2}u_{2,2} \dots u_{n,2} \dots u_{1,k}u_{2,k} \dots u_{n,k}$ with $k \geq 0$, $u_{i,j} \in A^*$, such that $u_{i,1}u_{i,2} \dots u_{i,k} = u_i$ for $1 \leq i \leq n$ [21]. We carry this notation over to pairs $(u, w) \in A^* \times B^*$ satisfying $|u| = |w|$, considering that such entities can be viewed as words in $(A \times B)^*$.

3 Decomposed Systems

We now introduce our notion of ‘decomposition into independent components’. Components are defined as concrete sub-systems of the respective system.

Let S be a system. A system c is a *sub-system* of S iff

1. $S_c \subseteq S_S$,
2. $s_c^i = s_S^i$,
3. $T_c \subseteq T_S$,
4. $\rightarrow_c = \rightarrow_S \cap (S_c \times T_c \times S_c)$,
5. $I_c = I_S \cap (T_c \times T_c)$, and
6. $l_c = l_S \upharpoonright_{T_c}$.

Let c_1 and c_2 be two sub-systems of S . We say c_1 and c_2 are *independent* (with respect to S), written $c_1 I_S c_2$, iff $T_{c_1} I_S T_{c_2}$. The *empty sub-system* of S is defined by $c_{empty}^S = (\{s_S^i\}, s_S^i, \emptyset, \emptyset, \emptyset, \emptyset)$.

Definition 2. A decomposition of a system S is a set $\mathcal{D} = \{c_1, \dots, c_n\}$, $n \in \mathbf{N}$, of sub-systems of S such that

1. $\forall i, j \in [1, n]. (i \neq j \implies c_i I_S c_j)$, and
2. $Runs(S) = \bigcup \{r_1 \otimes \dots \otimes r_n \mid r_i \in Runs(c_i) \text{ for all } i \in [1, n]\}$.

A decomposed system is a pair (S, \mathcal{D}) , where \mathcal{D} is a decomposition of system S .

Every system S has at least one decomposition: the one consisting of S itself. A system may well have many different decompositions: e.g., $P = a.0 \parallel b.0 \parallel c.0$ can be decomposed into $\{(a.0 \parallel b.0), c.0\}$, into $\{a.0, (b.0 \parallel c.0)\}$, and into $\{a.0, b.0, c.0\}$. Every non-empty system will, however, uniquely decompose into a set of *prime components*.

Definition 3. A sub-system c of a system S is a *divisor* of S iff there exists a decomposition \mathcal{D} of S such that $c \in \mathcal{D}$. A system S is *prime* iff S is non-empty, and c_{empty}^S and S are the only divisors of S .

Theorem 1. Each non-empty system S has a unique decomposition \mathcal{D} such that for all $c \in \mathcal{D}$ c is prime.

Proof (Sketch). This can be established following the standard proof of unique prime factorization of natural numbers (see e.g. [22]). Instead of proceeding by induction on \mathbf{N} , we proceed by induction on the smallest upper bound on the number of transitions that can occur concurrently at the initial state. This is possible due to our restriction to concurrency-degree finite systems. (c.f. Appendix C)

Definition 4. We define the prime components of a system S , denoted by $PComps(S)$, as follows: if S is empty we set $PComps(S) = \emptyset$, otherwise we define $PComps(S)$ to be the decomposition associated with S by Theorem 1.

Theorem 2. Let S be a finite system. $PComps(S)$ is computable.

Proof (Sketch). Let S be a non-empty finite system. We partition T_S into non-empty subsets such that each subset is a connected component with respect to the dependence relation (the complement of I_S). The sub-systems naturally induced by these sets of transitions are prime and together they form a decomposition of S . (c.f. Appendix C)

Convention 1. In the context of a decomposed system (S, \mathcal{D}) we use the following decomposition functions: $K : T_S \rightarrow \mathcal{D}$, defined by $K(t) = c_i \iff t \in T_{c_i}$, and $Ks : T_S^* \rightarrow \mathcal{P}(\mathcal{D})$, defined by $Ks(w) = \{K(t) \mid t \in w\}$. (K is a function by clause (1) of the definition of decomposition, and the irreflexivity of independence.)

If it is clear from the context that a system S is non-empty and there is no other decomposition specified, we understand S as the decomposed system $S = (S, PComps(S))$.

4 Composition

Hp, hhp, and chhp bisimilarity are composable with respect to decompositions in the following sense: whenever we can exhibit a one-to-one correspondence between the components of two decomposed systems such that related components are hp (hhp, chhp) bisimilar then the two systems are hp (hhp, chhp) bisimilar.

Theorem 3. Let $x \in \{hp, hhp, chhp\}$; let (S_1, \mathcal{D}_1) and (S_2, \mathcal{D}_2) be two decomposed systems. If there exists a bijection $\beta : \mathcal{D}_1 \rightarrow \mathcal{D}_2$ such that $c_1 \sim_x \beta(c_1)$ for each $c_1 \in \mathcal{D}_1$ then we have $S_1 \sim_x S_2$.

Proof (Sketch). Let (S_1, \mathcal{D}_1) and (S_2, \mathcal{D}_2) be two decomposed systems. Assume we are given a bijection $\beta : \mathcal{D}_1 \rightarrow \mathcal{D}_2$, say $\beta = \{(c_1^1, c_2^1), \dots, (c_1^n, c_2^n)\}$, and a family $\{\mathcal{H}^i\}_{i=1}^n$ such that for all $i \in [1, n]$ \mathcal{H}^i is a hp bisimulation relating c_1^i and c_2^i . We define $\mathcal{H} = \bigcup \{r^1 \otimes \dots \otimes r^n \mid r^i \in \mathcal{H}^i \text{ for all } i \in [1, n]\}$. It is straightforward to check that \mathcal{H} is a hp bisimulation relating S_1 and S_2 . Furthermore, it is routine to establish: if for all $i \in [1, n]$ \mathcal{H}^i is hereditary then \mathcal{H} will also be hereditary; if for all $i \in [1, n]$ \mathcal{H}^i is coherent then \mathcal{H} will also be coherent.

5 Decomposition

It is trivial that hp, hhp, and chhp bisimilarity are *not* decomposable in the converse sense: as we saw $P = a.0 \parallel b.0 \parallel c.0$ can be decomposed into $\{(a.0 \parallel b.0), c.0\}$ and also into $\{a.0, (b.0 \parallel c.0)\}$; but certainly we cannot exhibit a bijection between the two decompositions such that related components are bisimilar. The more natural question to ask is whether a notion of equivalence is decomposable with respect to *prime decompositions*.

The example of Figure 1 demonstrates hp bisimilarity is *not* decomposable in this sense, either. On the one hand, A and B are hp bisimilar. The additional transition b'_3 in B can easily be hidden by adopting the following strategy: if b'_3 occurs as the first transition we will match it against b_1 . Then in both systems ‘parallel b ’ is the only remaining behaviour, and b'_1 can safely be matched by b_2 . If we start out with b'_1 we will match it against b_1 . Then the a -transition is disabled in both systems, and this time it will be safe to match b'_3 by b_2 . On the other hand, a bijection between the prime components of A and those of B can clearly not be found.

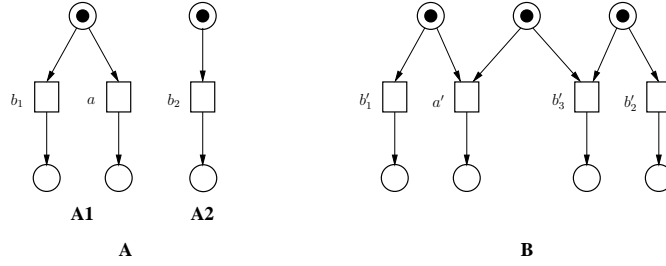


Fig. 1. The transitions of A and B are labelled as their names suggest: e.g. $l(b'_1) = b$. A consists of two prime components: A_1 and A_2 ; B has only one prime component: B itself

A and B are *not* (c)hhp bisimilar: at $(b_1 b_2, b'_1 b'_3)$ we can backtrack (b_1, b'_1) ; then the a -transition becomes available in A but not in B . In Section 7 we will briefly discuss whether (c)hhp bisimilarity may be decomposable with respect to prime decompositions. Here we want to analyse whether there are conditions under which we do obtain decomposition for hp bisimilarity; this is important with respect to establishing coincidence results. We will find that, on systems whose prime components are, what we shall call, *concurrent step connected* (*csc*), hp, and also hhp and chhp, bisimilarity are indeed decomposable with respect to prime decompositions: whenever two *csc-decomposable systems* are hp (hhp, chhp) bisimilar then there is a one-to-one correspondence between their prime components such that related components are hp (hhp, chhp) bisimilar.

We start out by explaining two special types of runs, which will play a key role in the proof. A run r is a *concurrent step* iff all the transitions on r occur independently of each other. A run r is *maximal with respect to initial concurrency* iff whenever a further transition t is executed at r , t will occur causally dependent on some transition on r .

Definition 5. Let S be a system, and $r \in \text{Runs}(S)$.

r is a concurrent step of S , written $r \in \text{csteps}(S)$, iff we have:

$$\forall k, l \in [1, |r|]. (k \neq l \Rightarrow r[k] \perp_S r[l]).$$

r is maximal with respect to initial concurrency, written $r \in \text{icmax}(S)$, iff we have:

$$\forall t \in T_S. (rt \in \text{Runs}(S) \Rightarrow \exists i \in [1, |r|]. i <_{rt} |rt|).$$

Clearly, in pairs of synchronous runs, and hence in hp bisimilarity, concurrent steps are always matched against concurrent steps.

Fact 1. Let S_1 and S_2 be two systems. For all $(r_1, r_2) \in SRuns(S_1, S_2)$ we have: $r_1 \in csteps(S_1) \iff r_2 \in csteps(S_2)$.

With the concept ‘maximal with respect to initial concurrency’ it is easy to identify a scenario which, given two decomposed systems $(S_1, \mathcal{D}_1), (S_2, \mathcal{D}_2)$, allows us to infer that two components $c_1 \in \mathcal{D}_1, c_2 \in \mathcal{D}_2$ are hp (hhp, chhp) bisimilar:

Lemma 1. Let $x \in \{hp, hhp, chhp\}$; let $(S_1, \mathcal{D}_1), (S_2, \mathcal{D}_2)$ be two decomposed systems. For any pair $c_1 \in \mathcal{D}_1, c_2 \in \mathcal{D}_2$ we have: if there exists $(r_1, r_2) \in \sim_x$ such that for $i = 1$, and 2 $\left\{ \begin{array}{l} c_i \notin Ks(r_i), \text{ and} \\ \forall c'_i \in \mathcal{D}_i \setminus c_i. r_i \uparrow c'_i \in icmax(c'_i) \end{array} \right\}$ then $c_1 \sim_x c_2$.

Proof (Sketch). Given entities as above, we can extract a hp (hhp, chhp) bisimulation relating c_1 and c_2 from any hp (hhp, chhp) bisimulation containing (r_1, r_2) . This is so because: (1) the full behaviour of c_1 and c_2 has still to be matched at (r_1, r_2) , and (2) the causal dependencies will force that behaviour of c_1 has to be matched against behaviour of c_2 , and vice versa. (c.f. Appendix D)

From the example of Figure 1 it is clear that, given two hp bisimilar systems, we may never be in a position to apply this lemma. A and B are hp bisimilar but there is no $(r_1, r_2) \in \sim_{hp}$ such that, via Lemma 1, we can deduce $c_1 \sim_{hp} c_2$ for any $c_1 \in PComps(A), c_2 \in PComps(B)$: if B , the only prime component of B , is not contained in $Ks(r_1)$ then $(r_1, r_2) = (\varepsilon, \varepsilon)$; but we neither have $\varepsilon \in icmax(A_1)$ nor $\varepsilon \in icmax(A_2)$.

The scenario of Lemma 1 will, however, certainly be available if for the system class under study we can show: the matching in hp bisimilarity respects prime components in that: let $(r_1, r_2) \in \sim_{hp}$; if, in (r_1, r_2) , a transition of prime component c_1 is matched to a transition of prime component c_2 , then, in (r_1, r_2) , any other transition of c_1 is also matched to a transition of c_2 , and vice versa. Then, given $(r_1, r_2) \in \sim_{hp}$, r_1 is ‘maximal with respect to initial concurrency’ for all but one prime component c_1 such that $c_1 \notin Ks(r_1)$ iff the analogue is true for r_2 . On second thought, to guarantee the applicability of Lemma 1 it is sufficient to obtain that the matching of *concurrent steps* (rather than the matching of *all runs*) respects prime components: concurrent steps can be seen as the minimum to consider when we want to achieve maximality with respect to initial concurrency.

We now identify a system class, as large as intuitively possible, which naturally satisfies this criteria: *csc-decomposable systems*. They have the following characteristic: each of their prime components is *cstep connected (csc)* in that: whenever we have computed a concurrent step r and we compute one further concurrently enabled transition t then there is the possibility of computing a sequence of transitions w such that the last transition of w is causally dependent on t and some transition of r . In short we may say: every concurrent step has a causal link with any further concurrently enabled transition.

Definition 6. Let S be a system.

Let $r \in Runs(S)$, and $k, l \in [1, |r|]$. $w \in T_S^+$ is a causal link at r between the events k and l , denoted by $w \in clinks_S(r, k, l)$, iff we have:

$$rw \in Runs(S) \ \& \ k <_{rw} |rw| \ \& \ l <_{rw} |rw|.$$

S is cstep connected (csc) iff for all $r \in csteps(S)$ with $|r| \geq 1$ we have:

$$\forall t \in T_S. (rt \in csteps(S) \Rightarrow \exists k \in [1, |r|]. \exists w \in T_S^+. w \in clinks_S(rt, k, |rt|)).$$

S is csc-decomposable iff every prime component of S is csc. (Note that non-empty csc systems are always prime.)

Example 1. Consider Figure 1. B is not csc: we can do b'_1 , and then b'_3 , but there is no causal link between b'_1 and b'_3 . Sequential systems ($\neg(\exists s, s', t, t'. t I_S t' \ \& \ s \xrightarrow{tt'}_S s')$), such as A_1 and A_2 , and initially sequential systems ($\forall r \in csteps(S). |r| \leq 1$) are trivially csc.

Lemma 2. Let S_1 and S_2 be two csc-decomposable systems. For all $(r_1, r_2) \in \sim_{hp}$ such that $r_i \in csteps(S_i)$ for $i = 1$, or 2 equivalently (Fact 1), we have:

$$\forall k, l \in [1, |r_1|]. (K(r_1[k]) = K(r_1[l]) \iff K(r_2[k]) = K(r_2[l])).$$

Proof (Sketch). We proceed by induction on the length of two related concurrent steps. Let (r_1, r_2) be given as above. Assume, in (r_1, r_2) , a transition of prime component c_1 is matched to a transition of prime component c_2 , and we want to match a further concurrently enabled c_1 -transition, t_1 . There will be a causal link at $r_1 t_1$ between event $|r_1 t_1|$ and one of the previously matched c_1 -events. By induction hypothesis we can assume these are all matched by c_2 -events. But then t_1 has to be matched by a c_2 -transition: otherwise the causal link could not be matched in a partial order preserving fashion. (c.f. Appendix D)

It is routine to derive the following corollaries:

Corollary 1. Let S_1 and S_2 be two csc-decomposable systems.

1. For all $(r_1, r_2) \in \sim_{hp}$ such that $r_i \in csteps(S_i)$ for $i = 1$, or 2 equivalently (Fact 1), we have: $|Ks(r_1)| = |Ks(r_2)|$.
2. If $S_1 \sim_{hp} S_2$ then $|PComps(S_1)| = |PComps(S_2)|$.

Corollary 2. Let S_1 and S_2 be two csc-decomposable systems, and let $(r_1, r_2) \in \sim_{hp}$ such that $r_i \in csteps(S_i)$ for $i = 1$, or 2 equivalently (Fact 1). For any pair of components $c_1 \in PComps(S_1)$, $c_2 \in PComps(S_2)$ such that $K(r_1[k]) = c_1$ and $K(r_2[k]) = c_2$ for some $k \in [1, |r_1|]$ we have: $r_1 \uparrow c_1 \in icmax(c_1) \iff r_2 \uparrow c_2 \in icmax(c_2)$.

For hhp and chhp bisimilarity there is now a simple argument that proves, for csc-decomposable systems, the two bisimilarities are indeed decomposable with respect to prime decompositions (c.f. Appendix D, Argument 2). This argument relies on backtracking; considering hp bisimilarity it is only obvious that, given two csc-decomposable systems S_1, S_2 with $S_1 \sim_{hp} S_2$, a bijection between $PComps(S_1)$ and $PComps(S_2)$ exists, and further, for each $c_1 \in PComps(S_1)$ there is $c_2 \in PComps(S_2)$ such that $c_1 \sim_{hp} c_2$, and vice versa. To prove decomposition for hp bisimilarity we need something more sophisticated: the combinatorial argument of Hall's Marriage Theorem (e.g. see [23]).

Theorem 4. Let $x \in \{hp, hhp, chhp\}$; let S_1, S_2 be two csc-decomposable systems. If $S_1 \sim_x S_2$ then there exists a bijection $\beta : PComps(S_1) \rightarrow PComps(S_2)$ between the prime components of S_1 and those of S_2 such that $c_1 \sim_x \beta(c_1)$ for each $c_1 \in PComps(S_1)$.

Proof. Let x, S_1, S_2 be given as above, and assume $S_1 \sim_x S_2$. We shall prove that a bijection β exists as required. By Corollary 1(2) we have (A) $|PComps(S_1)| = |PComps(S_2)|$, and it only remains to show that an injective map can be found. For each $c_1 \in PComps(S_1)$ let C_{2,c_1} be the set of prime components of S_2 which are x bisimilar to c_1 . By Hall's Marriage Theorem the required injection exists if and only if the following condition is fulfilled:

$$(*) \quad \forall C_1 \subseteq PComps(S_1). \quad \left| \bigcup_{c_1 \in C_1} C_{2,c_1} \right| \geq |C_1|.$$

Choose an arbitrary subset C_1 of $PComps(S_1)$. Let $\bar{C}_1 = PComps(S_1) \setminus C_1$, and consider $r_1 \in csteps(S_1)$ such that (B) $Ks(r_1) = \bar{C}_1$, and $\forall c_1 \in \bar{C}_1. r_1 \uparrow c_1 \in icmax(c_1)$; this is clearly possible. There must be r_2 such that $(r_1, r_2) \in \sim_x$; set $\bar{C}_2 = Ks(r_2)$, and $C_2 = PComps(S_2) \setminus \bar{C}_2$. By Corollary 2 we obtain $\forall c_2 \in \bar{C}_2. r_2 \uparrow c_2 \in icmax(c_2)$. On the other hand, (B) and Corollary 1(1) give us $|\bar{C}_1| = |\bar{C}_2|$, and considering (A) we gain (C) $|C_1| = |C_2|$. Next we show that for each remaining component $c_2 \in C_2$ there is a component $c_1 \in C_1$ such that $c_1 \sim_x c_2$. With (C) this will immediately establish $\left| \bigcup_{c_1 \in C_1} C_{2,c_1} \right| \geq |C_1|$, and thereby (*).

Assume C_2 is non-empty, and choose any $c_2 \in C_2$. Consider r'_2 such that $r_2 r'_2 \in csteps(S_2)$, $Ks(r'_2) = C_2 \setminus c_2$, and $\forall c'_2 \in C_2 \setminus c_2. r'_2 \uparrow c'_2 \in icmax(c'_2)$; this is clearly possible. Note that altogether we have (D) $Ks(r_2 r'_2) = PComps(S_2) \setminus c_2$, and $\forall c'_2 \in PComps(S_2) \setminus c_2. r_2 r'_2 \uparrow c'_2 \in icmax(c'_2)$. There must be r'_1 such that $(r_1 r'_1, r_2 r'_2) \in \sim_x$. Corollary 1(1) gives us $|Ks(r_1 r'_1)| = |Ks(r_2 r'_2)|$, and by (D), (A), and (B) this implies $Ks(r_1 r'_1) = PComps(S_1) \setminus c_1$ for some $c_1 \in C_1$. By Corollary 2 we obtain $\forall c'_1 \in PComps(S_1) \setminus c_1. r_1 r'_1 \uparrow c'_1 \in icmax(c'_1)$. But altogether this means we can apply Lemma 1 to infer $c_1 \sim_x c_2$. Thus, c_1 provides a component exactly as required.

6 Coincidence Results

We now apply our composition and decomposition theory to prove several coincidence results on hp, hhp, and chhp bisimilarity. First of all, our theory gives us a general proof technique: whenever we consider whether (any two of) the three equivalences coincide for a class of csc-decomposable systems, we can restrict our attention to the respective class of prime components. This is immediate by the following argument:

Argument 1. Assume two csc-decomposable systems S_1 and S_2 that are hp bisimilar. By Theorem 4(hp) we obtain a bijection between the prime components of S_1 and those of S_2 such that related components are hp bisimilar. Then, provided that hp, hhp, and chhp bisimilarity coincide for the class of the prime components, by Theorem 3(chhp) we can conclude that S_1 and S_2 are chhp (and thus also hhp) bisimilar.

It is folklore that for sequential systems hp, hhp, and chhp bisimilarity all coincide with classical bisimilarity (e.g. see [13]). Furthermore, we have already mentioned that sequential systems are csc. Then, with the previous argument we obtain:

Theorem 5. *Hp, hhp, and chhp bisimilarity coincide for parallel compositions of sequential systems. (Formally, a parallel composition of sequential systems is a system which can be decomposed into sequential components.)*

Consider the following generalization of the class ‘parallel compositions of sequential systems’: each system S is a parallel composition of *initially sequential* components such that each component may, by performing a transition, fork into a parallel composition of initially sequential sub-components, each of which may in turn evolve into a parallel composition of initially sequential sub-components, and so on; this description is complete in that we do not allow any communication between parallel threads. This system class is best known as, and most conveniently captured by, *communication-free net systems*³. (Formally, a net system \mathcal{N} is *communication-free* iff $\forall t \in T_{\mathcal{N}}. |\bullet t| = 1$.)

If a communication-free net system S is *concurrency-degree bounded* in that the smallest upper bound on the number of transitions that can be concurrently enabled in S with respect to any state, $cbound(S)$, is given by a natural number, then, for each proper component c of S , $cbound(c)$ will be strictly smaller than $cbound(S)$. With Argument 1 we then obtain coincidence for concurrency-degree bounded communication-free net systems by induction on $cbound(S)$ (c.f. Appendix E).

Definition 7. *Let S be a system. The smallest upper bound on the number of transitions that can be concurrently enabled in S with respect to any state, $cbound(S)$, is defined by $\max\{cbound_S(s) \mid s \in S_S\}$. S is concurrency-degree bounded iff $cbound(S) \in \mathbf{N}_0$.*

Theorem 6. *Two concurrency-degree bounded communication-free net systems are hp bisimilar iff they are hhp bisimilar iff they are chhp bisimilar.*

By translating Argument 1 into a tableau system, we achieve coincidence for *simple basic parallel processes (SBPP)*. These can be interpreted as an orthogonal class of communication-free net systems³: we lift the restriction to concurrency-degree bounded systems, but require our systems to be finitely representable. Following [7], SBPP are defined by process expressions of the grammar: $E ::= S \mid E \parallel E$, where ‘ \parallel ’ is parallel composition and S is an *initially sequential process* expression given by: $S ::= \mathbf{0} \mid a.E \mid S + S \mid X$, where $\mathbf{0}$ is the empty process, $a.E$, where $a \in Act$, is action prefix, ‘+’ is nondeterministic choice, and X is an ‘initially sequential process’ variable. Every SBPP can effectively be transformed into a chhp bisimilar SBPP in normal form.

Definition 8. *Let $Vars = \{X_1, X_2, \dots\}$ be a set of process variables, and $Vars^{\otimes} = \{\alpha, \beta, \dots\}$ the set of finite multisets over $Vars$. We identify $\alpha = \{X, X, Y\}$ with the parallel composition $X \parallel X \parallel Y$; the empty multiset is recognized as the process $\mathbf{0}$. A SBPP in normal form is a pair $\mathcal{E} = (E_0, \Delta_{\mathcal{E}})$, where $E_0 \in Vars^{\otimes}$, and $\Delta_{\mathcal{E}}$ is a finite family of recursive equations $\{X_i := E_i \mid 1 \leq i \leq m\}$. The X_i are distinct, and the E_i are of the form: $a_1.\alpha_1 + a_2.\alpha_2 + \dots + a_n.\alpha_n$, where $n \geq 1$, and $\forall i \in [1, n]. \alpha_i \in Vars^{\otimes}$. Further, $\forall i \in [0, m], E_i$ at most contains the variables $\{X_1, \dots, X_m\}$.*

³ As their unfoldings communication-free net systems also capture the class of communication-free weighted Petri nets.

Theorem 7. *Two SBPP are hp bisimilar iff they are hhp bisimilar iff they are chhp bisimilar.*

Proof (Sketch). The tableau proof system of Figure 2 gives rise to a decision procedure that decides whether two SBPP in normal form are hp bisimilar, and at the same time, whether they are chhp bisimilar. Rule **Match** provides matching for initially sequential processes; rule **Decomp** reflects our decomposition theory, and provides the means to reduce pairs of processes to check into smaller pairs of processes to compare. Theorem 4(hp) implies forward soundness of **Decomp** for hp bisimilarity, Theorem 3(chhp) gives us backwards soundness of **Decomp** for chhp bisimilarity. Finiteness, completeness for hp bisimilarity, and soundness for chhp bisimilarity of the tableau system can then be proved by using the standard arguments. (c.f. Appendix E.1)

$$\begin{array}{l}
 \mathbf{Rec} \quad \frac{X = Y}{E = F} \quad \text{where } (X := E) \in \Delta_{\mathcal{E}}, (Y := F) \in \Delta_{\mathcal{F}} \\
 \\
 \mathbf{Match} \quad \frac{\sum_{i=1}^n a_i \cdot \alpha_i = \sum_{j=1}^m b_j \cdot \beta_j}{\{\alpha_i = \beta_{f(i)}\}_{i=1}^n \quad \{\alpha_{g(j)} = \beta_j\}_{j=1}^m} \\
 \quad \text{where } f : [1, n] \rightarrow [1, m], g : [1, m] \rightarrow [1, n] \text{ are functions such that} \\
 \quad \forall i \in [1, n]. a_i = b_{f(i)}, \text{ and similarly for } g. \\
 \\
 \mathbf{Decomp} \quad \frac{\alpha = \beta}{\{X = Y\}_{(X, Y) \in b}} \quad \text{where } b : \alpha \rightarrow \beta \text{ is a bijection (relating variable instances).}
 \end{array}$$

A node n is a *successful terminal* iff

$n: \mathbf{0} = \mathbf{0}$, or

$n: X = Y$, and there is a node $n_a: X = Y$ above n in the tableau.

A node n is an *unsuccessful terminal* iff

$n: \alpha = \beta$, and a bijection b as required by rule **Decomp** does not exist, or

$n: \sum_{i=1}^n a_i \cdot \alpha_i = \sum_{j=1}^m b_j \cdot \beta_j$, and f and g as required by rule **Match** do not exist.

Fig. 2. A tableau system with respect to two SBPP in normal form \mathcal{E} and \mathcal{F}

7 Conclusions

There are further applications of our decomposition theory. In analogy to Argument 1 decidability of hp (hhp, chhp) bisimilarity on a class of finite-state csc-decomposable systems reduces to decidability on the respective class of prime components (recall Theorem 2). Further, if a system is specified in terms of csc components, our decomposition theory is profitable with respect to tackling the state explosion problem: we do not need to check hp (hhp, chhp) bisimilarity on the global state space but we can proceed by checking the respective equivalence on pairs of components.

One might speculate that (c)hhp bisimilarity is decomposable with respect to prime decompositions for systems in general: with the help of backtracking one might be able to prove a general version of Lemma 2; though this may be hard, or at least technically tedious, to carry through. Furthermore, as pointed out to me by Lasota, in the formulation of a general version of Lemma 2 and the decomposition theorem, one will have to address the issue of (c)hhp bisimilar choices: let $P = (P_1 \parallel P_2) + (P_1 \parallel P_2)$ and $Q = P_1 \parallel P_2$; clearly $P \sim_{(c)hhp} Q$ but since P is prime there is no bijection between the prime components of P and those of Q .

It is, of course, also possible to investigate whether a truly-concurrent equivalence satisfies the unique decomposition property usually investigated in the interleaving setting. (Given some class of process terms, is each of them uniquely, up to the equivalence, represented as a parallel composition of primes?) Indeed, unique decomposition with respect to distributed bisimilarity has been proved for BPP [24]. Note, however, that decomposition in this sense is *not* sufficient to establish the results of Section 6.

We hope this paper motivates the particular significance of composition and decomposition for true-concurrency: decomposition characteristics of a system class may translate into truly-concurrent equivalences or logics in a very concrete way, and thereby lead us to decision procedures and/or coincidence results. In this spirit, the ideas of the paper can be taken further: one could investigate whether a similar approach is possible with respect to temporal logics, and, orthogonally, whether our decomposition theory can be generalized by integrating a concept of synchronization. Indeed, the latter idea stands behind the result that (c)hhp bisimilarity is decidable for a class of live free-choice systems [13]. This is so far the only positive result on hhp bisimilarity for a class that admits a flexible form of synchronization. ([25] presents that hhp bisimilarity is decidable for trace-labelled systems but the proof turned out to be incomplete [15].)

Acknowledgements. I would like to thank Walter Vogler: he has provided crucial help by pointing out to me that Hall's Marriage Theorem has to be applied in the proof of an earlier version of Theorem 4. I thank Javier Esparza, Mogens Nielsen, Damian Niwinski, and the anonymous referees for their valuable comments on this work. I thank Monika Maidl, who has helped to clarify a question related to Theorem 2. Finally, I would like to thank Slawomir Lasota for pointing out to me the issue of (c)hhp bisimilar choices.

References

1. Penczek, W., Kuiper, R.: Traces and logic. In: *The Book of Traces*. World Scientific (1995) 307–381
2. Madhusudan, P., Thiagarajan, P.S.: Controllers for discrete event systems via morphisms. In: *CONCUR'98*. Volume 1466 of LNCS. (1998) 18–33
3. Jurdziński, M., Nielsen, M., Srba, J.: Domino hereditary history preserving bisimilarity is undecidable. *Inform. and Comput.* **184** (2003) 343–368
4. Vogler, W.: Deciding history preserving bisimilarity. In: *ICALP'91*. Volume 510 of LNCS. (1991) 495–505
5. Jategaonkar, L., Meyer, A.R.: Deciding true concurrency equivalences on safe, finite nets. *TCS* **154** (1996) 107–143
6. Montanari, U., Pistore, M.: Minimal transition systems for history-preserving bisimulation. In: *STACS'97*. Volume 1200 of LNCS. (1997) 413–425

7. Esparza, J., Kiehn, A.: On the model checking problem for branching time logics and basic parallel processes. In: CAV'95. Volume 939 of LNCS. (1995) 353–366
8. Sunesen, K., Nielsen, M.: Behavioural equivalence for infinite systems—partially decidable! In: ICATPN'96. Volume 1091 of LNCS. (1996) 460–479
9. Jančar, P.: Strong bisimilarity on basic parallel processes is PSPACE-complete. In: LICS'03, IEEE (2003) 216–??.
10. Srba, J.: Strong bisimilarity and regularity of basic parallel processes is PSPACE-hard. In: STACS'02. Volume 2285 of LNCS. (2002) 535–546
11. Lasota, S.: A polynomial-time algorithm for deciding true concurrency equivalences of basic parallel processes. In: MFCS'03. Volume 2747 of LNCS. (2003) 521–530
12. Fröschle, S.: Decidability of plain and hereditary history-preserving bisimulation for BPP. In: EXPRESS'99. Volume 27 of ENTCS. (1999)
13. Fröschle, S.: Decidability and Coincidence of Equivalences for Concurrency. PhD thesis, University of Edinburgh (2004)
14. Winskel, G., Nielsen, M.: Models for concurrency. In: Handbook of logic in computer science, Vol. 4. Oxford Univ. Press (1995) 1–148
15. Fröschle, S.: The decidability border of hereditary history preserving bisimilarity. Information Processing Letters (to appear)
16. Hirshfeld, Y., Jerrum, M., Moller, F.: A polynomial-time algorithm for deciding bisimulation equivalence of normed basic parallel processes. Mathematical Structures in Computer Science **6** (1996) 251–259
17. Jančar, P., Kot, M.: Bisimilarity on normed basic parallel processes can be decided in time $o(n^3)$. In: AVIS'04. ENTCS (2004)
18. Christensen, S., Hirshfeld, Y., Moller, F.: Decomposability, decidability and axiomatisability for bisimulation equivalence on basic parallel processes. In: LICS'93, IEEE (1993) 386–396
19. Milner, R., Moller, F.: Unique decomposition of processes. TCS **107** (1993) 357–363
20. Fröschle, S., Hildebrandt, T.: On plain and hereditary history-preserving bisimulation. In: MFCS'99. Volume 1672 of LNCS. (1999) 354–365
21. Pin, J.E.: Syntactic semigroups. In: Handbook of formal languages, Vol. 1. Springer (1997) 680–746
22. Norman, C.W.: Undergraduate Algebra. Oxford Science Publications (1986)
23. Truss, J.K.: Discrete Mathematics for Computer Scientists. Addison-Wesley (1991)
24. Christensen, S.: Decidability and Decomposition in Process Algebras. PhD thesis, University of Edinburgh (1993)
25. Mukund, M.: Hereditary history preserving bisimulation is decidable for trace-labelled systems. In: FST TCS'02. Volume 2556 of LNCS. (2002) 289–300

A Petri Net Terminology

A.1 Net Systems

A *labelled net* N is a tuple (P_N, T_N, F_N, l_N) , where P_N is the set of places, T_N is the set of transitions, $F_N : (P_N \times T_N) \cup (T_N \times P_N) \rightarrow \{0, 1\}$ is the flow relation, and $l_N : T_N \rightarrow Act$ is the labelling function, where Act is a set of actions. The *pre-set* of an element $x \in P_N \cup T_N$, $\bullet x$, is defined by $\{y \mid F_N(y, x) = 1\}$, the *post-set* of x , $x\bullet$, similarly is $\{y \mid F_N(x, y) = 1\}$.

A *marking* M of N is a map $P_N \rightarrow \mathbf{N}_0$. M *enables* a transition $t \in T_N$, denoted by $M[t]$, if $M(p) \geq F(p, t)$ for every $p \in P_N$. If $M[t]$ then t can occur. The resulting

marking M' is defined by $M'(p) := M(p) - F(p, t) + F(t, p)$ for all $p \in P_N$. Altogether we write $M \xrightarrow{t} M'$. Extending this notation to sequences of transitions, we write $M \xrightarrow{w} M'$, where $w = t_1 t_2 \dots t_n \in T_N^*$, to denote $M \xrightarrow{t_1} M_1 \xrightarrow{t_2} \dots \xrightarrow{t_n} M_n$ for some markings M_1, \dots, M_n with $M_n = M'$.

A *labelled Petri net* \mathcal{N} is a pair (N, M_0) , where N is a labelled net, and M_0 is the *initial marking* of \mathcal{N} . A marking M is *reachable* in \mathcal{N} , $M \in \text{Reach}(\mathcal{N})$, iff $M_0 \xrightarrow{w} M$ for some $w \in T_N^*$. \mathcal{N} is *1-safe* iff $\forall M \in \text{Reach}(\mathcal{N}). \forall p \in P_N. M(p) \leq 1$.

We refer to *1-safe labelled Petri nets* as *net systems*. We employ the usual graphical representation of Petri nets, where transitions are depicted as boxes, places as circles, and markings as tokens in the respective places.

We associate the following independence relation with a net system N : two transitions $t, t' \in T_N$ are *independent* in N , denoted by $t I_N t'$, iff their neighbourhoods of places do not intersect, i.e. iff $(\bullet t \cup t \bullet) \cap (\bullet t' \cup t' \bullet) = \emptyset$. Then we can indeed interpret net systems as a class of lats:

Fact 2. *Let \mathcal{N} be a net system. $(\text{Reach}(\mathcal{N}), M_0, T'_N, I_N \cap (T'_N \times T'_N), \rightarrow_N, l_N \upharpoonright_{T'_N})$ is a lat, where $T'_N = \{t \in T_N \mid \exists M \in \text{Reach}(\mathcal{N}). M \xrightarrow{t} \bullet\}$, and \rightarrow_N is the transition relation induced by the Petri net firing rule.*

A.2 Weighted Petri Nets and their Unfoldings

In view of Appendix E.1 we introduce some more standard Petri net definitions; we present them similarly to [7].

A *(labelled) weighted net* is a tuple (S, T, W, l) , where S and T are disjoint sets of places and transitions, $W : (S \times T) \cup (T \times S) \rightarrow \mathbf{N}_0$ is a *weight function*, and $l : T \rightarrow \text{Act}$ is a labelling function. The pre-set of an element $x \in S \cup T$, $\bullet x$, is defined by $\{y \in S \cup T \mid W(y, x) > 0\}$, the post-set of x , $x \bullet$, similarly is $\{y \in S \cup T \mid W(x, y) > 0\}$. A *weighted Petri net* is a pair (N, M_0) , where N is a weighted net and M_0 is a marking of N ; markings are defined as for unweighted nets.

Let (P, T, F, l) be a (non-weighted) net and let $x_1, x_2 \in P \cup T$. We say x_1 and x_2 are in *conflict*, denoted by $x_1 \# x_2$, if there exist distinct transitions $t_1, t_2 \in T$ 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 . $x \in P \cup T$ is in *self-conflict* iff $x \# x$. (This definition of conflict is not intuitive for general nets, but we will only use it in the context of *occurrence nets*.)

A *(labelled) occurrence net* 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.

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 $B \cup E$ with respect to \leq_N by $\text{Min}(N)$.

Let $\mathcal{N} = (N, M_0)$ be a weighted Petri net with $N = (S, T, W, l)$. A *branching process* of \mathcal{N} is a pair $\beta = (N', f)$, where $N' = (B, E, F, l')$ is an occurrence net, and f is a function $f : B \cup E \rightarrow S \cup T$ satisfying:

1. $f(B) \subseteq S$, and $f(E) \subseteq T$.
2. For all $t \in T, e \in E$ such that $f(e) = t$ we have: $\forall s \in S. |\bullet e \cap f^{-1}(s)| = W(s, t)$
& $|e \bullet \cap f^{-1}(s)| = W(t, s)$.
3. For all $e_1, e_2 \in E$ we have: $\bullet e_1 = \bullet e_2$ & $f(e_1) = f(e_2) \implies e_1 = e_2$.
4. $\forall s \in S. |\text{Min}(N') \cap f^{-1}(s)| = M_0(s)$.
5. $l' = l \circ f \upharpoonright_E$.

Let $\beta_1 = (N_1, f_1), \beta_2 = (N_2, f_2)$ be two branching processes of \mathcal{N} . An *isomorphism* from β_1 to β_2 is an isomorphism h from N_1 to N_2 such that $f_2 \circ h = f_1$ (or equally $f_1 \circ h^{-1} = f_2$). Branching processes are only considered up to isomorphism.

The set of branching processes of a (weighted) Petri net \mathcal{N} , denoted by $BP(\mathcal{N})$, is naturally structured by the following partial order: let $\beta_1, \beta_2 \in BP(\mathcal{N})$. $\beta_1 \leq \beta_2$ iff β_1 is an initial part of β_2 , and thus unfolds \mathcal{N} to a lesser degree than β_2 . $(BP(\mathcal{N}), \leq)$ forms a complete lattice. Then, each (weighted) Petri net \mathcal{N} has a largest branching process, the *unfolding* of \mathcal{N} , denoted by $unf(\mathcal{N})$.

B Bisimulation Approximations

In view of Appendix E.1 we give an alternative definition of ((c)h)hp bisimilarity based on bisimulation approximations.

Definition 9. Let S_1, S_2 be two systems, and $n \in \mathbf{N}_0$. A prefix-closed set $\mathcal{H} \subseteq SRuns(S_1, S_2)$ is a hp bisimulation approximation of degree n for S_1 and S_2 if:

1. $(\varepsilon, \varepsilon) \in \mathcal{H}$.
2. If $(r_1, r_2) \in \mathcal{H}$, $|r_1| < n$, and $r_1 t_1 \in Runs(S_1)$ for some $t_1 \in T_1$, then there is $t_2 \in T_2$ such that $(r_1 t_1, r_2 t_2) \in \mathcal{H}$.
3. Vice versa.

\mathcal{H} is a hhp bisimulation approximation of degree n when it further satisfies:

4. If $(r_1 t_1 w_1, r_2 t_2 w_2) \in \mathcal{H}$ for some $w_1 \in T_1^*, w_2 \in T_2^*, t_1 \in T_1$, and $t_2 \in T_2$ such that $|w_1| = |w_2|$, $t_1 I_1 w_1$ (or $t_2 I_2 w_2$ equivalently), then $(r_1 w_1, r_2 w_2) \in \mathcal{H}$.

\mathcal{H} is a chhp bisimulation approximation of degree n when it further satisfies:

5. If $(r_1 w_1, r_2 w_2), (r_1 t_1, r_2 t_2) \in \mathcal{H}$ for some $w_1 \in T_1^*, w_2 \in T_2^*, t_1 \in T_1$, and $t_2 \in T_2$ such that $|w_1| = |w_2|$, $|r_1 w_1| < n$, $t_1 I_1 w_1$, and $t_2 I_2 w_2$, then $(r_1 t_1 w_1, r_2 t_2 w_2) \in \mathcal{H}$.

We write $S_1 \sim_{((c)h)hp}^n S_2$ iff there is a ((c)h)hp bisimulation approximation of degree n relating them.

Lemma 3. For image-finite systems (which include concurrency-degree finite systems),

$$\sim_{((c)h)hp} = \bigcap_{n=0}^{\infty} \sim_{((c)h)hp}^n.$$

C Relating to Section 3

For the full proofs of Theorem 1 and 2 we require a further definition.

Definition 10. *Let S be a system.*

The sub-system of S induced by $T \subseteq T_S$ is defined by:

$$(S_T, s_S^i, T', \rightarrow_S \cap (S_T \times T' \times S_T), I_S \cap (T' \times T'), l_S \upharpoonright_{T'}),$$

where $S_T = \{s \in S_S \mid \exists w. s_S^i \xrightarrow{w} s \ \& \ (\forall j \in [1, |w|]. w[j] \in T)\}$,

and $T' = \{t \in T \mid \exists s, s' \in S_T. s \xrightarrow{t} s'\}$.

If c is a divisor of S we define $S \setminus c$ to be the sub-system of S induced by $T_S \setminus T_c$.

For the proof of Theorem 1 we also need the following observations on decompositions.

Proposition 1. *Let $(S, \{c_1, \dots, c_n\})$ be a decomposed system.*

1. *If C_i is a decomposition of c_i then $\{c_1, \dots, c_{i-1}, c_{i+1}, \dots, c_n\} \cup C_i$ is a decomposition of S , for all $i \in [1, n]$.*
2. *$\{c_1, \dots, c_{i-1}, c_{i+1}, \dots, c_n\}$ is a decomposition of $S \setminus c_i$, for all $i \in [1, n]$.*
3. *Let p be a divisor of S . If p is prime then p must be a divisor of c_i for some $i \in [1, n]$.*

Proof. (1) and (2) follow easily from the definitions. (3) seems more involved but this is also straightforward if one considers that divisors are defined as concrete sub-systems of the respective system: with the help of the definitions and the two axioms of independence (c.f. definition of lats', axiom (3) and (4)) one can show that p must be a divisor of the c_i that satisfies $T_p \cap T_{c_i} \neq \emptyset$ (such c_i must clearly exist).

Proof (Theorem 1). Let S be a non-empty system. Denote the smallest upper bound on the number of transitions that can occur concurrently at the initial state by $icb(S)$ ($= cbound_S(s_S^i)$). Since we only consider concurrency-degree finite systems $icb(S) \in \mathbf{N}_0$ (c.f. Section 2). We prove the fact by induction on $icb(S)$, say k .

If $k = 0$ the fact vacuously holds: this case is only possible for empty systems. Assume $k > 0$ and suppose the fact is true for systems S' with $icb(S') \leq k - 1$. First we show that S is decomposable into a set of primes. If S is prime this is immediate. If S is not prime then there must be a decomposition of S , say $\mathcal{D} = \{c_1, \dots, c_n\}$, such that $n > 1$, and for all $i \in [1, n]$ c_i is non-empty. This means for all $i \in [1, n]$ we must have $icb(c_i) < k$. But then by the induction hypothesis each c_i is decomposable into a set of primes, and so must be S (Prop. 1(1)).

Secondly, we prove the uniqueness of the factorization. Suppose S can be decomposed into $\{b_1, \dots, b_n\}$, and $\{c_1, \dots, c_m\}$ respectively, where $b_1, \dots, b_n, c_1, \dots, c_m$ are prime. By Prop. 1(3) b_1 divides one of the factors in the decomposition $\{c_1, \dots, c_m\}$. Say b_1 is a divisor of c_1 . Since c_1 is prime we conclude $b_1 = c_1$, and so $\{b_2, \dots, b_n\}$ and $\{c_2, \dots, c_m\}$ are decompositions of $S \setminus b_1$ (Prop 1(2)). But clearly $icb(S \setminus b_1) < k$, and thus by the induction hypothesis we can assume that $n = m$ and (possibly on renumbering c_1, \dots, c_m) $b_2 = c_2, \dots, b_n = c_m$. Hence the two prime decompositions of S are identical.

Proof (Theorem 2). Given a finite system S , we construct a set C of sub-systems of S in three steps, which are clearly computable.

1. We partition T_S into non-empty subsets T_1, \dots, T_n such that each T_i is a connected component with respect to the dependence relation D_S (the complement of I_S), that is two transitions t, t' belong to the same T_i iff $t D_S t_1 D_S \dots D_S t_m D_S t'$ for some $t_1, \dots, t_m \in T_S$.
2. For $i \in [1, n]$ we compute c_i , the sub-system of S induced by T_i .
3. We define $C = \{c_i\}_{i \in [1, n]}$.

We claim that $C = PComps(S)$. If S is empty then $C = \emptyset$ as required. Assume S is non-empty. First, we check that C is a decomposition of S . By definition it is clear that for all distinct $c, c' \in C, c I_S c'$. For $i \in [1, n]$ let $r_i \in Runs(c_i)$. Clearly $\forall i \in [1, n], r_i \in Runs(S)$. Then, by repeated use of the second axiom of independence (definition of lats', axiom (4)) we obtain $r_1 \otimes \dots \otimes r_n \subseteq Runs(S)$. Conversely, let $r \in Runs(S)$. Clearly $r \in r \upharpoonright_{T_1} \otimes \dots \otimes r \upharpoonright_{T_n}$. Further, by repeated use of the first axiom of independence (definition of lats', axiom (3)) we obtain $\forall i \in [1, n], r \upharpoonright_{T_i} \in Runs(c_i)$. To see that each $c_i \in C$ is prime consider that by definition each c_i does not contain any independent factors.

D Relating to Section 5

Fact 3. *Let (S, \mathcal{D}) be a decomposed system, let $r \in Runs(S)$.*

1. *Let $i, j \in [1, |r|]$. If $i <_r j$ then $r[i]$ and $r[j]$ belong to the same component.*
2. *Let $c \in \mathcal{D}$, and $r_c \in Runs(c)$ with $r \upharpoonright c = r_c$. If $t_c \in T_c$ can occur in c at r_c , it can also occur in S at r , and the causal dependencies of t_c with respect to r will exactly reflect the causal dependencies of t_c with respect to r_c .*

Proof (Lemma 1). Let $(S_1, \mathcal{D}_1), (S_2, \mathcal{D}_2)$ be two decomposed systems. Let $c_1 \in \mathcal{D}_1, c_2 \in \mathcal{D}_2$, and $(r_1, r_2) \in SRuns(S_1, S_2)$ such that $\left\{ \begin{array}{l} c_i \notin Ks(r_i), \text{ and} \\ \forall c'_i \in \mathcal{D}_i \setminus c_i. r_i \upharpoonright c'_i \in icmax(c'_i) \end{array} \right\}$ for $i = 1, 2$. Consider the following statement:

$$(*) \text{ For all } (r'_1, r'_2) \text{ such that } (r_1 r'_1, r_2 r'_2) \in \sim_{hp} \text{ we have:} \\ \forall k \in [1, |r'_1|]. (K(r'_1[k]) = c_1 \iff K(r'_2[k]) = c_2).$$

Given a hp bisimulation \mathcal{H} for S_1 and S_2 such that $(r_1, r_2) \in \mathcal{H}$, define $\mathcal{H}_c = \{(r'_1 \upharpoonright c_1, r'_2 \upharpoonright c_2) \mid (r_1 r'_1, r_2 r'_2) \in \mathcal{H}\}$. Assuming (*) it is easy to check that \mathcal{H}_c is a hp bisimulation relating c_1 and c_2 : the full behaviour of c_1 and c_2 has still to be matched at (r_1, r_2) ; by (*) behaviour of c_1 will be matched against behaviour of c_2 , and vice versa. It is also routine to verify: if \mathcal{H} is hereditary then \mathcal{H}_c will be hereditary; if \mathcal{H} is hereditary and coherent then \mathcal{H}_c will be hereditary and coherent.

We shall now prove that (*) indeed holds. We proceed by induction on the length of r'_1 , or r'_2 equivalently. Base case $r'_1 = r'_2 = \varepsilon$: There is nothing to prove. Inductive case $r'_1 = r''_1 t_1$ and $r'_2 = r''_2 t_2$: By prefix-closure of \sim_{hp} we can apply the induction hypothesis to (r''_1, r''_2) . Thus, we only need to show: $K(t_1) = c_1 \iff K(t_2) = c_2$.

Suppose $K(t_1) = c_1$. We prove $K(t_2) = c_2$ by case analysis. First, assume there is $k \in [1, |r'_1|]$ such that $k <_{r'_1} |r'_1|$. Considering Fact 3(1) this implies $K(r'_1[k]) = c_1$. By induction hypothesis we further obtain $K(r'_2[k]) = c_2$. Since \sim_{hp} is partial order

preserving, we have $k <_{r'_2} |r'_1|$. But then, again considering Fact 3(1), we can infer $K(t_2) = c_2$ as required. If there is no k as above then it is easy to derive $r_1 t_1 \in csteps(S_1)$, and further $r_2 t_2 \in csteps(S_2)$. Since for all $c'_2 \in \mathcal{D}_2 \setminus c_2$ we assume $r_2 \uparrow c'_2 \in icmax(c'_2)$ it is then immediate that $K(t_2) = c_2$. The opposite direction follows from the symmetric argument.

Proof (Lemma 2). Let S_1 and S_2 be two csc-decomposable systems, and $(r_1, r_2) \in \sim_{hp}$ such that $r_i \in csteps(S_i)$ for $i = 1, 2$ equivalently (Fact 1). We prove

$$(*) \forall k, l \in [1, |r_1|]. (K(r_1[k]) = K(r_1[l]) \iff K(r_2[k]) = K(r_2[l]))$$

by induction on the length of r_1 , or r_2 equivalently.

Base case $r_1 = r_2 = \varepsilon$: There is nothing to prove. Inductive case $r_1 = r'_1 t_1$ and $r_2 = r'_2 t_2$: By prefix-closure of \sim_{hp} and the induction hypothesis we can assume that $(*)$ holds for (r'_1, r'_2) . Thus, we only have to prove: $\forall k \in [1, |r'_1|]. (K(t_1) = K(r_1[k]) \iff K(t_2) = K(r_2[k]))$. We establish this by *reductio ad absurdum*.

Set $c_1 = K(t_1)$. Assume there is $k \in [1, |r'_1|]$ such that (a) $K(r_1[k]) = c_1$ but (b) $K(r_2[k]) \neq K(t_2)$. Set $r'_{c_1} = r'_1 \uparrow c_1$. Clearly, $r_1 \uparrow c_1 = r'_{c_1} t_1$. Further, we have $r'_{c_1}, r'_{c_1} t_1 \in csteps(c_1)$. Then, since c_1 is csc there exist $l \in [1, |r'_{c_1}|]$ and $w_1 \in T_{c_1}^+$ such that $w_1 \in clinks_{c_1}(r'_{c_1} t_1, l, |r'_{c_1} t_1|)$. Considering Fact 3(2), we infer $w_1 \in clinks_{S_1}(r_1, m, |r_1|)$, where m satisfies (c) $m \in [1, |r'_1|]$ & $K(r_1[m]) = c_1$. There must be a match for w_1 at (r_1, r_2) : there must be w_2 such that $(r_1 w_1, r_2 w_2) \in \sim_{hp}$. Since \sim_{hp} is partial order preserving we obtain $w_2 \in clinks_{S_2}(r_2, m, |r_1|)$. But this leads to a contradiction. On the one hand, considering Fact 3(1), we infer $K(r_2[m]) = K(t_2)$. On the other hand, by $k \in [1, |r'_1|]$, (a), (c), and the induction hypothesis we obtain $K(r_2[k]) = K(r_2[m])$, and further with (b), $K(r_2[m]) \neq K(t_2)$. The other direction follows from the symmetric argument.

Argument 2. Let S_1 and S_2 be two csc-decomposable systems that are (c)hhp bisimilar. Choose a concurrent step r_1 of S_1 such that for each $c_1 \in PComps(S_1)$, $r_1 \uparrow c_1 \in icmax(c_1)$; this is clearly possible. There must be r_2 such that $(r_1, r_2) \in \sim_{(c)hhp}$. By Lemma 2 this match induces an injective map β from $PComps(S_1)$ to $PComps(S_2)$; by Corollary 1(2) β must be bijective. Corollary 2 implies for each $c_2 \in PComps(S_2)$, $r_2 \uparrow c_2 \in icmax(c_2)$. But then we obtain $c_1 \sim_{(c)hhp} c_2$ for each pair $(c_1, c_2) \in \beta$ as follows: at (r_1, r_2) backtrack all the transitions of c_1 and c_2 ; then apply Lemma 1.

E Relating to Section 6

Proof (Theorem 6). Let S_1, S_2 be two concurrency-degree bounded communication-free net systems such that $S_1 \sim_{hp} S_2$. We proceed by induction on $cbound(S_1)$ ($= cbound(S_2)$), say n . The case $n = 0$ is trivial: S_1 and S_2 are both empty.

If $n = 1$ then S_1 and S_2 are either both sequential, or they are both initially sequential and will fork later on. In the first case $S_1 \sim_{chhp} S_2$ is clearly given since hp and chhp bisimilarity coincide for sequential systems. In the latter case we can copy matchings for the sequential beginning from any hp bisimulation relating S_1 and S_2 , and then we proceed as for $n > 1$; thereby we will obtain $S_1 \sim_{chhp} S_2$.

If $n > 1$ we can apply the induction hypothesis to the prime components of S_1 and those of S_2 , and $S_1 \sim_{chhp} S_2$ follows by Argument 1.

E.1 SBPP

To give a full proof of Theorem 7 we first of all need to define partial order semantics for SBPP so that ((c)h)hp bisimilarity on SBPP obtains a formal meaning. There is a standard way to do so: SBPP correspond to BPP in normal form, and these are well-known to correspond to communication-free weighted Petri nets. Partial order semantics for weighted Petri nets is in turn provided by their unfoldings. For the definition of weighted Petri nets and their unfoldings see Appendix A.2. Our approach is similar to that of [7].

Definition 11. A weighted net $N = (S, T, W, l)$ is communication-free iff $\forall t \in T. |\bullet t| = 1 \ \& \ \forall s \in S, t \in T. W(s, t) \leq 1$. A weighted Petri net $\mathcal{N} = (N, M_0)$ is communication-free iff N is communication-free.

Observe that every SBPP can effectively be transformed into a normal form representative (Def. 8) by using operations that only affect the appearance of the defining expressions, that is ‘syntactic’ operations like introduction of new variables, substitution of variables for subexpressions, and unfolding of variables. Thus, SBPP in normal form indeed represent the entire SBPP class.

Let $\mathcal{E} = (E_0, \Delta)$ be a SBPP in normal form; w.l.o.g. assume $E_0 = \alpha$ for some $\alpha \in \text{Vars}^\otimes$. The translation from \mathcal{E} to a communication-free weighted Petri net goes as follows. The *net representation* of Δ , denoted by $\text{net}(\Delta)$, is defined by the tuple (S, T, W, l) , where $S = \text{Vars}(\Delta)$, the variables that occur in Δ , and T and W are given as follows: if $\langle X := E \rangle \in \Delta$, and $a_i.\alpha_i$ is a subexpression of E , then we introduce a transition t , labelled by a_i , with $W(X, t) = 1$, and $W(t, Y) = m$, where Y appears m times in α_i . The *Petri net representation* of \mathcal{E} , denoted by $PN(\mathcal{E})$, is defined by the pair $(\text{net}(\Delta), M_0)$, where M_0 is such that $\forall X \in \text{Vars}(\Delta). M_0(X) = |\alpha|_X$.

The *unfolding* of \mathcal{E} , denoted by $\text{unf}(\mathcal{E})$, is defined by $\text{unf}(PN(\mathcal{E}))$. Sometimes, we shall identify $\text{unf}(\mathcal{E})$ with the Petri net $(N, \text{Min}(N))$, where (N, f) is a representative of $\text{unf}(\mathcal{E})$.

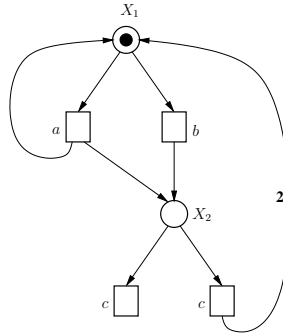


Fig. 3. The Petri net representation of \mathcal{E}

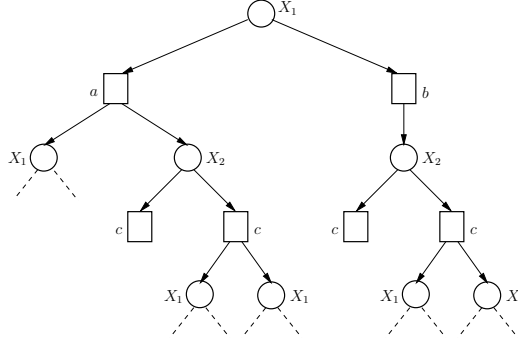


Fig. 4. The unfolding of $PN(\mathcal{E})$, and thus \mathcal{E}

Example 2. Figure 3 gives the Petri net representation of the SBPP $\mathcal{E} = (\Delta, X_1)$, where $\Delta = \{X_1 := a.(X_1 \parallel X_2) + b.X_2; X_2 := c.\mathbf{0} + c.(X_1 \parallel X_1)\}$. Figure 4 demonstrates the unfolding of $PN(\mathcal{E})$, and thus \mathcal{E} .

It is easy to see that, viewed as a Petri net, unfoldings of SBPP are 1-safe. Hence, ((c)h)hp bisimilarity for SBPP is formally defined as follows: two SBPP \mathcal{E} and \mathcal{F} are ((c)h)hp bisimilar iff $unf(\mathcal{E})$ and $unf(\mathcal{F})$ are ((c)h)hp bisimilar. (Recall from Appendix A.1 that net systems can be viewed as lats’.) The unfolding of a SBPP is, of course, communication-free (as defined for the 1-safe case in Section 6).

We now formally explain how the tableau system of Figure 2 gives rise to Theorem 7. First, we transfer our composition and decomposition theory to SBPP. We require decomposition for hp bisimilarity, and composition for chhp bisimilarity, or to be precise for chhp bisimilarity approximations (c.f. Appendix B).

Lemma 4. *Let \mathcal{E}, \mathcal{F} be two SBPP with $E_0 = \alpha, F_0 = \beta$.*

1. *If $\mathcal{E} \sim_{hp} \mathcal{F}$ then there exists a bijection $b : \alpha \rightarrow \beta$ (relating variable instances) such that $\forall X \in \alpha. X \sim_{hp} b(X)$.*
2. *If there exists a bijection $b : \alpha \rightarrow \beta$ (relating variable instances) such that $\forall X \in \alpha. X \sim_{chhp}^n b(X)$ then $\mathcal{E} \sim_{chhp}^n \mathcal{F}$.*

Proof. Each place of the initial marking of $unf(\mathcal{E})$, and hence each $X \in \alpha$, represents an initially sequential component; the thus defined components will never communicate with each other but run completely independently: they exactly correspond to the prime components of $unf(\mathcal{E})$, and since initially sequential they are csc. This is analogous for \mathcal{F} . Then it is immediate that (1) follows from Theorem 4(hp), and (2) follows from the analogue of Theorem 3(chhp) for approximations: it is routine to check that this is also valid.

This lemma provides the means of decomposing a goal of two processes to check into subgoals of “smaller” processes to test. Note however, that the lemma will not take us any further if the processes consist of one initially sequential component each, or,

in other words, if the processes correspond to sums. For this case we need a further insight:

Lemma 5. *Let \mathcal{E}, \mathcal{F} be two SBPP with $E_0 = \sum_{i=1}^n a_i \cdot \alpha_i$, $F_0 = \sum_{j=1}^m b_j \cdot \beta_j$.*

1. *If $\mathcal{E} \sim_{hp} \mathcal{F}$ then the following two conditions hold:*
 - (a) *For each $i \in [1, n]$ there is $j \in [1, m]$ such that $a_i = b_j$, and $\alpha_i \sim_{hp} \beta_j$.*
 - (b) *For each $j \in [1, m]$ there is $i \in [1, n]$ such that $a_i = b_j$, and $\alpha_i \sim_{hp} \beta_j$.*
2. *If the following two conditions hold then $\mathcal{E} \sim_{chhp}^{k+1} \mathcal{F}$:*
 - (a) *For each $i \in [1, n]$ there is $j \in [1, m]$ such that $a_i = b_j$, and $\alpha_i \sim_{chhp}^k \beta_j$.*
 - (b) *For each $j \in [1, m]$ there is $i \in [1, n]$ such that $a_i = b_j$, and $\alpha_i \sim_{chhp}^k \beta_j$.*

Proof. The a_i exactly correspond to the transitions enabled at $unf(\mathcal{E})$, and each α_i describes the behaviour that remains after executing a_i ; this is similar for $unf(\mathcal{F})$. Then (1) is an immediate consequence of the definition of (hp) bisimulation.

To verify (2) assume we are given label-preserving functions $f : [1, n] \rightarrow [1, m]$, $g : [1, m] \rightarrow [1, n]$, and a family of chhp bisimulation approximations of degree k , $\{\mathcal{H}_{(i,j)}\}_{(i,j) \in f \cup g}$, such that for each $(i, j) \in f \cup g$, $\mathcal{H}_{(i,j)}$ relates $unf(\alpha_i)$ and $unf(\beta_j)$. Each a_i and b_j corresponds to exactly one enabled transition of $unf(\mathcal{E})$, and $unf(\mathcal{F})$ respectively; we denote the respective transitions by $t_i^\mathcal{E}$, and $t_j^\mathcal{F}$ respectively. We define a relation \mathcal{H} as follows:

$$\mathcal{H} = \{(\varepsilon, \varepsilon)\} \cup \{(t_i^\mathcal{E} r^\mathcal{E}, t_j^\mathcal{F} r^\mathcal{F}) \mid (i, j) \in g \cup f \ \& \ (r^\mathcal{E}, r^\mathcal{F}) \in \mathcal{H}_{(i,j)}\}.$$

It is easy to check that \mathcal{H} is a chhp bisimulation approximation of degree $k + 1$ relating \mathcal{E} and \mathcal{F} (possibly on renaming transitions in the obvious way). In particular, to see that \mathcal{H} is partial order preserving, hereditary, and coherent consider: all enabled transitions of $unf(\mathcal{E})$, and $unf(\mathcal{F})$ respectively, are in conflict with each other; the remaining behaviour of $unf(\mathcal{E})$ after the occurrence of the transition $t_i^\mathcal{E}$ is dependent on the event corresponding to $t_i^\mathcal{E}$, and similarly for $unf(\mathcal{F})$.

Completeness for hp bisimilarity and soundness for chhp bisimilarity of the tableau system will follow from the first, and respectively second, part of Lemma 4 and 5. W.l.o.g. we assume that a tableau is started with an expression of the form $\alpha = \beta$. The typical order of rule instantiations will be: **Decomp**, **Rec**, **Match**, **Decomp**, ...; it will always be the case that the subgoals of a rule instantiation match the premise of the rule next in this sequence. We proceed in this manner until we hit one of the specified terminal nodes. Note how the second condition for successful terminals makes sure we ‘loop back’ whenever we encounter a pair of variables that we have already dealt with before. This will ensure finiteness of the tableau system.

Lemma 6 (finiteness). *Every tableau for two given SBPP in normal form is finite. Furthermore, for two given SBPP in normal form the number of possible tableaux is finite.*

Proof. Let \mathcal{E} with $\Delta_\mathcal{E} = \{X_i \stackrel{def}{=} E_i : i = 1, 2, \dots, n\}$, \mathcal{F} with $\Delta_\mathcal{F} = \{Y_j \stackrel{def}{=} F_j : j = 1, 2, \dots, m\}$ be two given SBPP in normal form. Assume to the contrary an infinite tableau $T(E_0 = F_0)$. Clearly, any tableau will be finite-branching. Then,

König's Lemma applies, and we can 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 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 $X = Y$ at our disposal, and after at most $m \times n$ instances of **Rec** we will 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} . So clearly, there can be only finitely many different tableaux for two SBPP.

Lemma 7 (completeness for hp bisimilarity). *Let \mathcal{E} and \mathcal{F} be two SBPP in normal form. If $\mathcal{E} \sim_{hp} \mathcal{F}$ then there exists a successful tableau $T(E_0 = F_0)$.*

Proof. Assume we are given two SBPP in normal form \mathcal{E} and \mathcal{F} such that $\mathcal{E} \sim_{hp} \mathcal{F}$. We shall show there exists a successful tableau $T(E_0 = F_0)$.

The tableau rules are forward sound in the following sense: if we apply a rule to a pair of hp bisimilar expressions, we can always find a rule instantiation such that the expressions related by the subgoals of the rule are hp bisimilar as well. This is obvious for rule **Rec**, and follows for **Decomp** from Lemma 4(1), and for **Match** from Lemma 5(1).

Thus, starting from the root we can build a tableau such that every node relates two expressions that are hp bisimilar. Since every tableau is finite, this construction will surely terminate. It follows from Lemma 4(1) and 5(1) that two expressions that are related by unsuccessful terminal nodes cannot be hp bisimilar, and so each terminal node will be successful. Hence, we have proved that there indeed exists a successful tableau.

Lemma 8 (soundness for chhp bisimilarity). *Let \mathcal{E} and \mathcal{F} be two SBPP in normal form. If there is a successful tableau $T(E_0 = F_0)$ then $\mathcal{E} \sim_{chhp} \mathcal{F}$.*

Proof. Let \mathcal{E}, \mathcal{F} be two SBPP in normal form. To the contrary assume there is a successful tableau $T(E_0 = F_0)$, but $\mathcal{E} \not\sim_{chhp} \mathcal{F}$. We shall show that this assumption leads to a contradiction.

If $\mathcal{E} \not\sim_{chhp} \mathcal{F}$ then by Lemma 3 there is a least k such that $\mathcal{E} \sim_{chhp}^n \mathcal{F}$ for all $n < k$ and $\mathcal{E} \not\sim_{chhp}^n \mathcal{F}$ for all $n \geq k$.

Note that the tableau rules are backwards sound w.r.t. \sim_{chhp}^n : if we have a rule instantiation such that the related expressions of each subgoal are chhp bisimilar of approximation n , then the expressions related by the premise must be chhp bisimilar of approximation n , as well. This is obvious for rule **Rec**, and follows for **Decomp** from Lemma 4(2). For **Match** we have a strengthening: the expressions related by the premise must be chhp bisimilar of approximation $n + 1$. This is a consequence of Lemma 5(2).

Thus, in our assumed tableau we can trace a path π such that $E \not\sim_{(k)chhp} F$ for the related expressions E and F of each node. While tracing this path we can mark each node with the least l such that $E \sim_{chhp}^n F$ for all $n < l$, and $E \not\sim_{chhp}^n F$ for all $n \geq l$. Note that the sequence of these measures along π is strictly decreasing due to instantiations of **Match**.

By finiteness (Lemma 6) π will end in a terminal node n_t . Since the tableau is successful n_t 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 $\mathbf{0} \sim_{chhp} \mathbf{0}$. 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.

Completeness for hp bisimilarity implies completeness for hhp and chhp bisimilarity, and similarly soundness for chhp bisimilarity gives soundness for hhp and hp bisimilarity. Then the decidability of hp, hhp, and chhp bisimilarity is straightforward: we only have to check whether there exists a successful tableau; by finiteness this can easily be done by exhaustive search. Since we decide the three bisimilarities by the same decision procedure it also follows that they coincide.