Hereditary History Preserving Bisimilarity Is Undecidable

Marcin Jurdziński[1] Mogens Nielsen[1]

BRICS[2]
Department of Computer Science
University of Aarhus

Abstract History preserving bisimilarity (hp-bisimilarity) and hereditary history preserving bisimilarity (hhp-bisimilarity) are behavioural equivalences taking into account causal relationships between events of concurrent systems. Their prominent feature is being preserved under action refinement, an operation important for the top-down design of concurrent systems. We show that—unlike hp-bisimilarity—checking hhp-bisimilarity for finite labelled asynchronous transition systems is not decidable, by a reduction from the halting problem of 2-counter machines. To make the proof more transparent we introduce an intermediate problem of checking domino bisimilarity for origin constrained tiling systems, whose undecidability is interesting in its own right. We also argue that the undecidability of hhp-bisimilarity holds for finite labelled 1-safe Petri nets.

1 Introduction

The notion of behavioural equivalence that has attracted most attention in concurrency theory is bisimilarity, originally introduced by Park [20] and Milner [15]; concurrent programs are considered to have the same meaning if they are bisimilar. The prominent role of bisimilarity is due to many pleasant properties it enjoys; we mention a few of them here.

A process of checking whether two transition systems are bisimilar can be seen as a two player game which is in fact an Ehrenfeucht-Fraïssé type of game for modal logic. More precisely, there is a winning strategy for a player who wants to show that the systems are bisimilar if and only if the systems cannot

[1] Address: BRICS, Department of Computer Science, University of Aarhus, Ny Munkegade, Building 540, 8000 Aarhus C, Denmark. Email: `mju@brics.dk`.

[2] **B**asic **R**esearch in **C**omputer **S**cience,
Centre of the Danish National Research Foundation.

be distinguished by the formulas of the logic; the result due to Hennessy and Milner [9].

Another notable property of bisimilarity is its computational feasibility; see for example the overview note [16]. Let us illustrate this on the examples of finite transition systems and a class of infinite-state transition systems generated by context free grammars. For finite transition systems there are very efficient polynomial time algorithms for checking bisimilarity [13,19], in sharp contrast to **PSPACE**-completeness of the classical language equivalence. For transition systems generated by context free grammars, while language equivalence is undecidable, bisimilarity is decidable [3], and if the grammar has no redundant nonterminals, even in polynomial time [10]. Furthermore, as the results of [8] indicate, bisimilarity has a very rare status of being a decidable equivalence for context free grammars: all the other equivalences in the linear/branching time hierarchy [6] are indeed undecidable. The algorithmic tractability makes bisimilarity especially attractive for automatic verification of concurrent systems.

The essence of bisimilarity, quoting [9], "is that the behaviour of a program is determined by how it communicates with an observer." Therefore, the notion of what can be observed of a behaviour of a system affects the notion of bisimilarity. An abstract definition of bisimilarity for arbitrary categories of models due to Joyal *et al.* [12] formalizes this idea. Given a category of models where objects are behaviours and morphisms correspond to extension of behaviours, and given a subcategory of observable behaviours, the abstract definition yields a notion of bisimilarity for all behaviours with respect to observable behaviours. For example, for rooted labelled transition systems, taking synchronization trees [15] into which they unfold as their behaviours, and sequences of actions as the observable behaviours, we recover the standard strong bisimilarity of Park and Milner [12].

In order to model concurrency more faithfully several models have been introduced (see [23] for a survey) that make explicit the distinction between events that can occur concurrently, and those that are causally related. Then a natural choice is to replace sequences, *i.e.*, linear orders as the observable behaviours, by partial orders of occurrences of events with causality as the ordering relation. For example, taking unfoldings of labelled asynchronous transition systems into event structures as the behaviours, and labelled partial orders as the observations, Joyal *et al.* [12] obtained from their abstract definition the hereditary history preserving bisimilarity (hhp-bisimilarity), independently introduced and studied by Bednarczyk [1].

A similar notion of bisimilarity has been studied before, namely history preserving bisimilarity (hp-bisimilarity), introduced by Rabinovich and Trakhtenbrot [21] and van Glabbeek and Goltz [7]. For the relationship between hp- and hhp-bisimilarity see for example [1,12,5].

One of the important motivations to study partial order based equivalences was the discovery that hp-bisimilarity has a rare status of being preserved under action refinement [7], an operation important for the top-down design of concurrent systems. Bednarczyk [1] has extended this result to hhp-bisimilarity.

There is a natural logical characterization of hhp-bisimilarity checking games as shown by Nielsen and Clausen [17]: they are characteristic games for an extension of modal logic with backwards modalities, interpreted over event structures.

Hp-bisimilarity has been shown to be decidable for 1-safe Petri nets by Vogler [22], and to be **DEXP**-complete by Jategaonkar, and Meyer [11]; let us just mention here that 1-safe Petri nets can be regarded as a proper subclass of finite asynchronous transition systems (see [23] for details), and that decidability of hp-bisimilarity can be easily extended to all finite asynchronous transition systems using the methods of [11].

Hhp-bisimilarity seems to be only a slight strengthening of hp-bisimilarity [12], and hence many attempts have been made to extend the above mentioned algorithms to the case of hhp-bisimilarity. However, decidability of hhp-bisimilarity has remained open, despite several attempts over the years [17,18,2,5]. Fröschle and Hildebrandt [5] have discovered an infinite hierarchy of bisimilarity notions refining hp-bisimilarity, and coarser than hhp-bisimilarity, such that hhp-bisimilarity is the intersection of all the bisimilarities in the hierarchy. They have shown all these bisimilarities to be decidable for 1-safe Petri nets. Fröschle [4] has shown hhp-bisimilarity to be decidable for BPP-processes, a class of infinite state systems.

In this paper, we finally settle the question of decidability of hhp-bisimilarity by showing it to be undecidable for finite 1-safe Petri nets. In order to make the proof more transparent we first introduce an intermediate problem of domino bisimilarity and show its undecidability by a direct reduction from the halting problem of 2-counter machines.

2 Hereditary history preserving bisimilarity

Definition 1 (Labelled asynchronous transition system)
A *labelled asynchronous transition system* is a tuple $A = (S, s^{\mathrm{ini}}, E, \to, L, \lambda, I)$, where S is its set of *states*, $s^{\mathrm{ini}} \in S$ is the *initial state*, E is the set of *events*, $\to \subseteq S \times E \times S$ is the set of *transitions*, L is the set of labels, and $\lambda : E \to L$ is the *labelling function*, and $I \subseteq E^2$ is the *independence relation* which is irreflexive and symmetric. We often write $s \xrightarrow{e} s'$, instead of $(s, e, s') \in \to$. Moreover, the following conditions have to be satisfied:

1. if $s \xrightarrow{e} s'$, and $s \xrightarrow{e} s''$, then $s' = s''$,
2. if $(e, e') \in I$, $s \xrightarrow{e} s'$, and $s' \xrightarrow{e'} t$, then $s \xrightarrow{e'} s''$, and $s'' \xrightarrow{e} t$ for some $s'' \in S$.

An asynchronous transition system is *coherent* if it satisfies one further condition:

3. if $(e, e') \in I$, $s \xrightarrow{e} s'$, and $s \xrightarrow{e'} s''$, then $s' \xrightarrow{e'} t$, and $s'' \xrightarrow{e} t$ for some $t \in S$.

[Definition 1] □

Winskel and Nielsen [23,18] give a thorough survey and establish formal relationships between asynchronous transition systems and other models for concurrency, such as Petri nets, and event structures. The independence relation is meant to model concurrency: independent events can occur concurrently, while those that are not independent are causally related or in conflict.

Let $A = (S, s^{\text{ini}}, E, \to, L, \lambda, I)$ be a labelled asynchronous transition system. A sequence of events $\overline{e} = \langle e_1, e_2, \ldots, e_n \rangle \in E^n$ is a *run* of A if there are states $s_1, s_2, \ldots, s_{n+1} \in S$, such that $s_1 = s^{\text{ini}}$, and for all $i \in [n]$ we have $s_i \xrightarrow{e_i} s_{i+1}$. We denote the set of runs of A by $\text{Runs}(A)$. We extend the labelling function λ to runs in the standard way. We say that $k \in [n]$ is *most recent* in $\overline{e}$, and we denote it by $k \in \text{MR}(\overline{e})$, if and only if $(e_k, e_\ell) \in I$ for all ℓ such that $k < \ell \leq n$. Note that if $k \in \text{MR}(\overline{e})$ then $\overline{e} \oslash k = \langle e_1, \ldots, e_{k-1}, e_{k+1}, \ldots, e_n \rangle \in \text{Runs}(A)$.

Definition 2 (Hereditary history preserving bisimulation)
Let $A_i = (S_i, s_i^{\text{ini}}, E_i, \to_i, L, \lambda_i, I_i)$ for $i \in \{1, 2\}$ be labelled asynchronous transition systems. A relation $B \subseteq \text{Runs}(A_1) \times \text{Runs}(A_2)$ is a *hereditary history preserving* (hhp-) bisimulation relating A_1 and A_2 if the following conditions are satisfied:

1. $(\varepsilon, \varepsilon) \in B$,

and if $(\overline{e_1}, \overline{e_2}) \in B$ then $\lambda_1(\overline{e_1}) = \lambda_2(\overline{e_2})$, and:

2. for all $e_1 \in E_1$, if $\overline{e_1} \cdot e_1 \in \text{Runs}(A_1)$, then there exists $e_2 \in E_2$, such that $\overline{e_2} \cdot e_2 \in \text{Runs}(A_2)$, and $\lambda_1(e_1) = \lambda_2(e_2)$, and $(\overline{e_1} \cdot e_1, \overline{e_2} \cdot e_2) \in B$,
3. for all $e_2 \in E_2$, if $\overline{e_2} \cdot e_2 \in \text{Runs}(A_2)$, then there exists $e_1 \in E_1$, such that $\overline{e_1} \cdot e_1 \in \text{Runs}(A_1)$, and $\lambda_1(e_1) = \lambda_2(e_2)$, and $(\overline{e_1} \cdot e_1, \overline{e_2} \cdot e_2) \in B$,
4. $k \in \text{MR}(\overline{e_1})$, if and only if $k \in \text{MR}(\overline{e_2})$,
5. if $k \in \text{MR}(\overline{e_1}) = \text{MR}(\overline{e_2})$, then $(\overline{e_1} \oslash k, \overline{e_2} \oslash k) \in B$. [Definition 2] □

Two asynchronous transition systems A_1, and A_2 are hereditary history preserving (hhp-) *bisimilar*, if there is an hhp-bisimulation relating them.

Remark 1 The term hereditary history preserving bisimulation originates from the fact that this notion of bisimulation has an alternative definition, which is formally a small strengthening of the standard definition of history preserving bisimulation [21,7], based explicitly on partial order behaviours [1,12]. Note that Definition 2 does not mention partial order behaviours explicitly, but they are implicit in the notion of most recent occurrences of events. For the proof of equivalence of our definition and the other ones see [17].

The main result of this paper is the following theorem proved in section 4.

Theorem 3 (Undecidability of hhp-bisimilarity)
Hhp-bisimilarity is undecidable for finite labelled asynchronous transition systems.

3 Domino bisimilarity is undecidable

3.1 Domino bisimilarity

Definition 4 (Origin constrained tiling system)
An *origin constrained tiling system* $T = \big(D, D^{\text{ori}}, (H, H^0), (V, V^0), L, \lambda\big)$ consists of a set D of *dominoes*, its subset $D^{\text{ori}} \subseteq D$ called the *origin constraint*, two *horizontal compatibility* relations $H, H^0 \subseteq D^2$, two *vertical compatibility* relations $V, V^0 \subseteq D^2$, a set L of *labels*, and a *labelling function* $\lambda : D \to L$.
[Definition 4] □

A *configuration* of T is a triple $(d, x, y) \in D \times \mathbb{N} \times \mathbb{N}$, such that if $x = y = 0$ then $d \in D^{\text{ori}}$. In other words, in the "origin" position $(x, y) = (0, 0)$ of the non-negative integer grid only dominoes from the origin constraint D^{ori} are allowed.

Let (d, x, y), and (d', x', y') be configurations of T such that $|x'-x|+|y'-y| = 1$, *i.e.*, the positions (x, y), and (x', y') are neighbours in the non-negative integer grid. Without loss of generality we may assume that $x + y < x' + y'$. We say that configurations (d, x, y), and (d', x', y') are *compatible* if either of the two conditions below holds:

- $x' = x$, and $y' = y + 1$, and
  if $y = 0$, then $(d, d') \in V^0$, and if $y > 0$, then $(d, d') \in V$, or
- $x' = x + 1$, and $y' = y$, and
  if $x = 0$, then $(d, d') \in H^0$, and if $x > 0$, then $(d, d') \in H$.

Definition 5 (Domino bisimulation)
Let $T_i = \big(D_i, D_i^{\text{ori}}, (H_i, H_i^0), (V_i, V_i^0), L_i, \lambda_i\big)$ for $i \in \{1, 2\}$ be origin constrained tiling systems. A relation $B \subseteq D_1 \times D_2 \times \mathbb{N} \times \mathbb{N}$ is a *domino bisimulation* relating T_1 and T_2, if $(d_1, d_2, x, y) \in B$ implies that $\lambda_1(d_1) = \lambda_2(d_2)$, and the following conditions are satisfied for all $i \in \{1, 2\}$:

1. for all $d_i \in D_i^{\text{ori}}$, there is $d_{3-i} \in D_{3-i}^{\text{ori}}$, so that $\lambda_1(d_1) = \lambda_2(d_2)$, and $(d_1, d_2, 0, 0) \in B$,
2. for all $x, y \in \mathbb{N}$, such that $(x, y) \neq (0, 0)$, and $d_i \in D_i$, there is $d_{3-i} \in D_{3-i}$, such that $\lambda_1(d_1) = \lambda_2(d_2)$, and $(d_1, d_2, x, y) \in B$,
3. if $(d_1, d_2, x, y) \in B$, then for all neighbours $(x', y') \in \mathbb{N} \times \mathbb{N}$ of (x, y), and $d_i' \in D_i$, if configurations (d_i, x, y), and (d_i', x', y') of T_i are compatible, then there exists $d_{3-i}' \in D_{3-i}$, such that $\lambda_1(d_1') = \lambda_2(d_2')$, and configurations (d_{3-i}, x, y), and (d_{3-i}', x', y') of T_{3-i} are compatible, and $(d_1', d_2', x', y') \in B$.

[Definition 5] □

We say that two tiling systems are *domino bisimilar* if and only if there is a domino bisimulation relating them.

Theorem 6 (Undecidability of domino bisimilarity)
Domino bisimilarity is undecidable for origin constrained tiling systems.

The proof is a reduction from the halting problem for deterministic 2-counter machines. For a deterministic 2-counter machine M we define in section 3.3 two origin constrained tiling systems T_1, and T_2, enjoying the following property.

Proposition 7 Machine M does not halt, if and only if there is a domino bisimulation relating T_1 and T_2.

3.2 Counter machines

A 2-counter machine M consists of a finite program with the set L of instruction labels, and instructions of the form:

- ℓ: $\mathtt{c}_i$ `:=` $\mathtt{c}_i$ `+ 1; goto` m
- ℓ: `if` $\mathtt{c}_i$ `= 0 then` $\mathtt{c}_i$ `:=` $\mathtt{c}_i$ `+ 1; goto` m
  `else` $\mathtt{c}_i$ `:=` $\mathtt{c}_i$ `- 1; goto` n
- `halt:`

where $i = 1, 2$; $\ell, m, n \in L$, and $\{\mathtt{start}, \mathtt{halt}\} \subseteq L$. A configuration of M is a triple $(\ell, x, y) \in L \times \mathbb{N} \times \mathbb{N}$, where ℓ is the label of the current instruction, and x, and y are the values stored in counters $\mathtt{c}_1$, and $\mathtt{c}_2$, respectively; we denote the set of configurations of M by $\mathrm{Confs}(M)$. The semantics of 2-counter machines is standard: let $\vdash_M \subseteq \mathrm{Confs}(M) \times \mathrm{Confs}(M)$ be the usual one-step derivation relation on configurations of M; by $\vdash_M^+$ we denote the reachability (in at least one step) relation for configurations, *i.e.*, the transitive closure of $\vdash_M$.

Before we give a reduction from the halting problem of 2-counter machines to origin constrained domino bisimilarity let us take a look at the directed graph $(\mathrm{Confs}(M), \vdash_M)$, with configurations of M as vertices, and edges denoting derivation in one step. Since machine M is deterministic, for each configuration there is at most one outgoing edge; moreover only halting configurations have no outgoing edges. It follows that connected components of the graph $(\mathrm{Confs}(M), \vdash_M)$ are either trees with edges going to the root which is the unique halting configuration in the component, or have no halting configuration at all. This observation implies the following proposition.

Proposition 8 Let M be a 2-counter machine. The following conditions are equivalent:

1. machine M halts on input $(0, 0)$, *i.e.*, $(\mathtt{start}, 0, 0) \vdash_M^+ (\mathtt{halt}, x, y)$ for some $x, y \in \mathbb{N}$,
2. $(\mathtt{start}, 0, 0) \sim_M (\mathtt{halt}, x, y)$ for some $x, y \in \mathbb{N}$, where the relation $\sim_M \subseteq \mathrm{Confs}(M) \times \mathrm{Confs}(M)$ is the symmetric and transitive closure of $\vdash_M$.

3.3 The reduction

Now we go for a proof of Proposition 7. The idea is to design a tiling system which "simulates" behaviour of a 2-counter machine.

Let M be a 2-counter machine. We construct a tiling system T_M with the set L of instruction labels of M as the set of dominoes, and the identity function on L as the labelling function. Note that this implies that all tuples belonging to a domino bisimulation relating copies of T_M are of the form (ℓ, ℓ, x, y), so we can identify them with configurations of M, *i.e.*, sometimes we will make no distinction between (ℓ, ℓ, x, y) and $(\ell, x, y) \in \mathrm{Confs}(M)$ for $\ell \in L$.

We define the horizontal compatibility relations $H_M, H_M^0 \subseteq L \times L$ of the tiling system T_M as follows:

– $(\ell, m) \in H_M$ if and only if either of the instructions below is an instruction of machine M:
  - ℓ: $\texttt{c}_1$ `:=` $\texttt{c}_1$ `+ 1; goto` m
  - m: `if` $\texttt{c}_1$ `= 0 then` $\texttt{c}_1$ `:=` $\texttt{c}_1$ `+ 1; goto` n
    `else` $\texttt{c}_1$ `:=` $\texttt{c}_1$ `- 1; goto` ℓ

– $(\ell, m) \in H_M^0$ if and only if $(\ell, m) \in H_M$, or the instruction below is an instruction of machine M:
  - ℓ: `if` $\texttt{c}_1$ `= 0 then` $\texttt{c}_1$ `:=` $\texttt{c}_1$ `+ 1; goto` m
    `else` $\texttt{c}_1$ `:=` $\texttt{c}_1$ `- 1; goto` n

Vertical compatibility relations V_M, and V_M^0 are defined in the same way, with $\texttt{c}_1$ instructions replaced with $\texttt{c}_2$ instructions. We also take $D_M^{\mathrm{ori}} = L$, *i.e.*, all dominoes are allowed in position $(0, 0)$. Note that the identity function is a 1-1 correspondence between configurations of M, and configurations of the tiling system T_M; from now on we will hence identify configurations of M and T_M. It follows immediately from the construction of T_M, that two configurations $c, c' \in \mathrm{Confs}(M)$ are compatible as configurations of T_M, if and only if $c \vdash_M c'$, or $c' \vdash_M c$, *i.e.*, compatibility relation of T_M coincides with the symmetric closure of $\vdash_M$. By $\approx_M$ we denote the symmetric and transitive closure of the compatibility relation of configurations of T_M. The following proposition is then straightforward.

Proposition 9 The two relations $\sim_M$, and $\approx_M$ coincide.

Now we are ready to define the two origin constrained tiling systems T_1, and T_2, postulated in Proposition 7. The idea is to have two independent and slightly pruned copies of T_M in T_2: one without the initial configuration $(\texttt{start}, 0, 0)$, and the other without any halting configurations $(\texttt{halt}, x, y)$. The other tiling system T_1 is going to have three independent copies of T_M: the two of T_2, and moreover, another full copy of T_M.

More formally we define $D_2 = \big(L \times \{1, 2\}\big) \setminus \big\{(\texttt{halt}, 2)\big\}$, and $D_2^{\mathrm{ori}} = D_2 \setminus \big\{(\texttt{start}, 1)\big\}$, and $V_2 = \big((V_M \otimes 1) \cup (V_M \otimes 2)\big) \cap (D_2 \times D_2)$, where for a binary relation R we define $R \otimes i$ to be the relation $\big\{ \big((a, i), (b, i)\big) \,:\, (a, b) \in R \big\}$. The other compatibility relations V_2^0, H_2, and H_2^0 are defined analogously from the respective compatibility relations of T_M.

The tiling system T_1 is obtained from T_2 by adding yet another independent copy of T_M, this time a complete one: $D_1 = D_2 \cup (L \times \{3\})$, and $D_1^{\mathrm{ori}} = D_2^{\mathrm{ori}} \cup$

$(L \times \{3\})$, and $V_1 = V_2 \cup (V_M \otimes 3)$, *etc.* The labelling functions of T_1, and T_2 are defined as $\lambda_i\big((\ell, i)\big) = \ell$.

In order to show Proposition 7, and hence conclude the proof of Theorem 6, it suffices to establish the following two claims.

Claim 10 If machine M halts on input $(0, 0)$, then there is no domino bisimulation relating T_1 and T_2.

Claim 11 If machine M does not halt on input $(0, 0)$, then there is a domino bisimulation relating T_1 and T_2.

4 Hhp-bisimilarity is undecidable

The proof of Theorem 3 is a reduction from the problem of deciding domino bisimilarity for origin constrained tiling systems. A method of encoding a tiling system on an infinite grid in the unfolding of a finite asynchronous transition system is due to Madhusudan and Thiagarajan [14]; we use a modified version of a gadget invented by them. For each origin constrained tiling system T we define an asynchronous transition system $A(T)$, such that the following proposition holds.

Proposition 12 There is a domino bisimulation relating origin constrained tiling systems T_1 and T_2, if and only if there is a hhp-bisimulation relating the asynchronous transition systems $A(T_1)$ and $A(T_2)$.

Let $T = \big(D, D^{\mathrm{ori}}, (H, H^0), (V, V^0), L, \lambda\big)$ be an origin constrained tiling system. We define the asynchronous transition system $A(T)$. The schematic structure of $A(T)$ can be seen in Figure 1. The set of events is defined as:

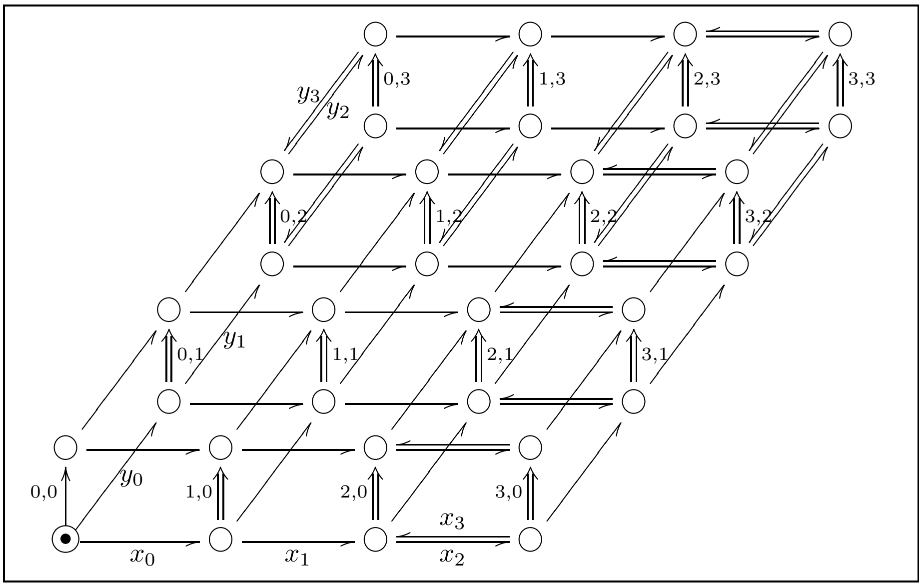
$$\begin{aligned} E_{A(T)} = {} & \big\{\, x_i, y_i \;:\; i \in \{0,1,2,3\} \,\big\} \\ & \cup \big\{\, d_{ij}, \overline{d}_{ij} \;:\; i, j \in \{0,1,2\}, d \in D, \text{ and } d \in D^{\mathrm{ori}} \text{ if } (i,j) = (0,0) \,\big\}. \end{aligned}$$

The rough idea behind the construction of $A(T)$ is best explained in terms of its event structure unfolding [23], in which the configurations of x- and y-transitions simply represent the grid structure of a tiling system, following [14]. Configurations in general consist of such a grid point plus at most two "d"- and "$\overline{d}$"-events, where the vertical (horizontal) compatibility of the tiling system is represented by the independence between a d_{ij}- and a $\overline{d}_{i(j+1)}$- ($\overline{d}_{(i+1)j}$-) event.

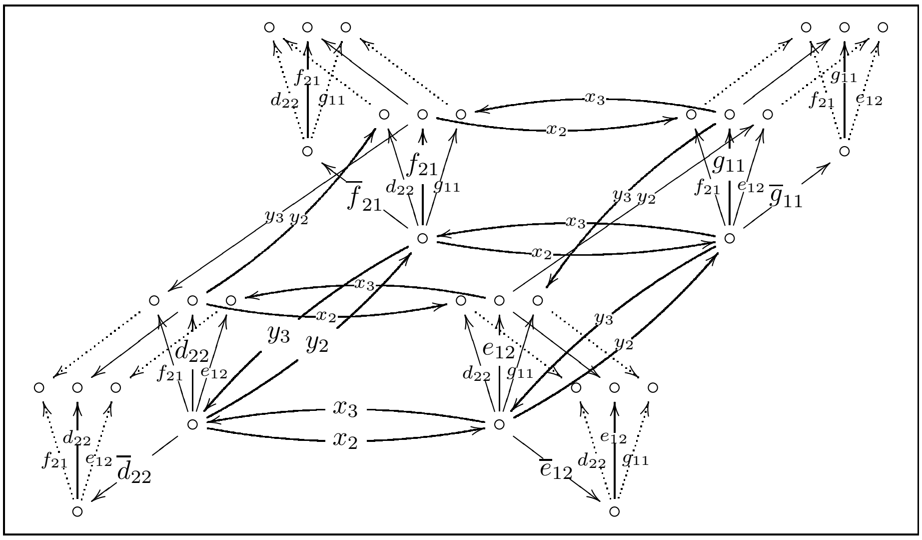
Notation: By abuse of notation we sometimes write d_{xy} or $\overline{d}_{xy}$ for $x, y \in \mathbb{N}$; we always mean by that the events $d_{\hat{x}\hat{y}}$ or $\overline{d}_{\hat{x}\hat{y}}$, respectively, where for $z \in \mathbb{N}$ we define $\hat{z}$ to be z if $z \leq 2$, and 2 for even z, and 1 for odd z if $z > 2$. [Notation] $\diamond$

The labelling function replaces dominoes in "d"-, and "$\overline{d}$"-events, with their labels in the tiling system:

$$\lambda_{A(T)}(e) = \begin{cases} e & \text{if } e \in \big\{\, x_i, y_i \;:\; i \in \{0, \ldots, 3\} \,\big\}, \\ \lambda(d)_{ij} & \text{if } e = d_{ij}, \text{ for some } d \in D, \\ \overline{\lambda(d)}_{ij} & \text{if } e = \overline{d}_{ij}, \text{ for some } d \in D. \end{cases}$$



(a) The structure of $A(T)$ in the large.



(b) The fine structure of the upper-right cube of $A(T)$.

Figure1. The structure of the asynchronous transition system $A(T)$.

The states, events, and transitions of $A(T)$ can be read from Figure 1; we briefly explain below how to do it.

There are sixteen states in the bottom layer of the structure in Figure 1(a). Let us identify these sixteen states with pairs of numbers shown on the vertical macro-arrows originating in these states shown in Figure 1(a). Each of these macro-arrows denotes a bundle of d_{ij}-, and $\overline{d}_{ij}$-event transitions sticking out of the state below, arranged in the fashion shown in Figure 1(b). For each state (i,j), and domino $d \in D$, there are d_{ij}-, and $\overline{d}_{ij}$-event transitions sticking out, and moreover for each state (i',j') from which there is an arrow in Figure 1(a) to state (i,j), there is a $d_{i'j'}$-event transition sticking out of (i,j). The state $(0,0)$ is exceptional: only dominoes from the origin constraint D^{ori} are allowed as events of transitions sticking out of it. It is also the initial state of $A(T)$.

As can be seen in Figure 1(b), from both ends of the d_{ij}-event transition rooted in state (i,j), there is an x_i-event transition to the corresponding (bottom, or top) $(i \oplus 1, j)$ state, and an y_i-event transition to the corresponding $(i, j \oplus 1)$ state, where $i \oplus 1 = i+1$ if $i < 3$, and $i \oplus 1 = 2$ if $i = 3$.

For each $d_{i'j'}$-event transition t sticking out of state (i,j), and each $e \in D$, there can be a pair of transitions which together with t and the $\overline{e}_{ij}$-event transition form a "diamond" of transitions; the events of the transitions lying on the opposite sides of the diamond coincide then. This type of transitions is shown in Figure 1(b) as dotted arrows. The condition for the two transitions closing the diamond to exist is that configurations (d,i',j') and $\big(e, i'+|i'-i|, j'+|j'-j|\big)$ of T are compatible, or $(i',j') = (i,j)$ and $e = d$. We define the independence relation $I_{A(T)} \subseteq E_{A(T)} \times E_{A(T)}$, to be the symmetric closure of the set:

$$
\begin{aligned}
&\big\{\ (x_i,y_j),(x_i,d_{ij}),(y_j,d_{ij})\ :\ i,j \in \{0,\dots,3\},\ \text{and } d \in D\ \big\}\ \cup\\
&\big\{\ (d_{ij},\overline{d}_{ij})\ :\ i,j \in \{0,1,2\},\ \text{and } d \in D\ \big\}\ \cup\\
&\big\{\ (d_{0j},\overline{e}_{1j})\ :\ j \in \{0,1,2\},\ \text{and } (d,e) \in H^0\ \big\}\ \cup\\
&\big\{\ (d_{ij},\overline{e}_{(i+1)j})\ :\ i \in \{1,2\}, j \in \{0,1,2\},\ \text{and } (d,e) \in H\ \big\}\ \cup\\
&\big\{\ (d_{i0},\overline{e}_{i1})\ :\ i \in \{0,1,2\},\ \text{and } (d,e) \in V^0\ \big\}\ \cup\\
&\big\{\ (d_{ij},\overline{e}_{i(j+1)})\ :\ i \in \{0,1,2\}, j \in \{1,2\},\ \text{and } (d,e) \in V\ \big\}.
\end{aligned}
$$

Note that it follows from the above that all diamonds of transitions in $A(T)$ are in fact independence diamonds.

Proof sketch (of Proposition 12)**:** The idea is to show that every domino bisimulation for T_1 and T_2 gives rise to an hhp-bisimulation for $A(T_1)$ and $A(T_2)$, and *vice versa*. First observe, that a run of $A(T_i)$ for $i \in \{1,2\}$ consists of a number of occurrences of x_j- and y_k-events, x and y of them respectively, and a set of "d"- and "$\overline{d}$"-events, which is of size at most two. In other words, we can map runs of $A(T_i)$ into triples (F_i, x, y), where $F_i \subseteq E_{A(T_i)}$ contains at most two "d"- and "$\overline{d}$"-events, and $x, y \in \mathbb{N}$. Define $\mathrm{Confs}\big(A(T_1), A(T_2)\big)$ to be the set of quadruples (F_1, F_2, x, y) where F_i's are as above and $x, y \in \mathbb{N}$. Then it is a matter of routine verification to see that there exists an hhp-bisimulation between $A(T_1)$ and $A(T_2)$, if and only if there exists a relation $B \subseteq \mathrm{Confs}\big(A(T_1), A(T_2)\big)$, such that $\{ (\overline{e_1}, \overline{e_2}) : (F_1, F_2, x, y) \in B, \text{where } \overline{e_i} \text{ is mapped to } (F_i, x, y) \}$ is an hhp-bisimulation relating $A(T_1)$ and $A(T_2)$. Hence, in the following we identify an hhp-bisimulation with such a relation B. The following claim immediately implies Proposition 12.

Claim 13 1. Let $B \subseteq \mathrm{Confs}\big(A(T_1), A(T_2)\big)$ be an hhp-bisimulation relating $A(T_1)$ and $A(T_2)$. Then the set $\{ (d, e, x, y) : \big(\{\overline{d}_{xy}\}, \{\overline{e}_{xy}\}, x, y\big) \in B \}$ is a domino bisimulation for T_1 and T_2.
2. Let $B \subseteq \mathrm{Confs}(T_1, T_2)$ be a domino bisimulation relating T_1 and T_2. Then the set $\{ \big(\{\overline{d}_{xy}\}, \{\overline{e}_{xy}\}, x, y\big) : (d, e, x, y) \in B \}$ can be extended to an hhp-bisimulation for $A(T_1)$ and $A(T_2)$.

This concludes the proof of Theorem 3. [Proposition 12] ■

As a corollary of the above proof we get the following strengthening of our main theorem.

Corollary 14 Hhp-bisimilarity is undecidable for finite labelled 1-safe Petri nets.

Proof sketch (of Corollary 14)**:** An attentive reader might have noticed, that the asynchronous transition system $A(T)$ as described in section 4, and sketched in Figure 1, is not coherent, while all asynchronous transition systems derived from (1-safe) Petri nets are [23,18]. It turns out, however, that $A(T)$ is not far from being coherent: it suffices to close all the diamonds with events d_{ij}, and x_i in positions $(i, j \oplus 1)$, and with events d_{ij}, and y_j in positions $(i \oplus 1, j)$, for $i, j \in \{0, \ldots, 3\}$; note that runs ending at the top of these diamonds are maximal runs. This completion of the transition structure of $A(T)$ does not affect the arguments used to establish Claim 13, and hence Theorem 3, but since it would obscure the picture in Figure 1(b), we have decided not to draw it there. It is laborious but routine to construct a 1-safe Petri net whose derived asynchronous transition system is isomorphic to the completion of $A(T)$ mentioned above.

[Corollary 14] ■

References

1. Marek A. Bednarczyk. Hereditary history preserving bisimulations or what is the power of the future perfect in program logics. Technical report, Polish Academy of Sciences, Gdańsk, April 1991. Available at `http://www.ipipan.gda.pl/~marek`.
2. Gian Luca Cattani and Vladimiro Sassone. Higher dimensional transition systems. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science*, pages 55–62, New Brunswick, New Jersey, 27–30 July 1996. IEEE Computer Society Press.
3. Søren Christensen, Hans Hüttel, and Colin Stirling. Bisimulation equivalence is decidable for all context-free processes. *Information and Computation*, 121(2):143–148, 1995.

4. Sibylle Fröschle. Decidability of plain and hereditary history-preserving bisimilarity for BPP. Presented at 6th International Workshop on Expressiveness in Concurrency, EXPRESS'99, Eindhoven, The Netherlands, August 1999.
5. Sibylle Fröschle and Thomas Hildebrandt. On plain and hereditary history-preserving bisimulation. In Mirosław Kutyłowski, Leszek Pacholski, and Tomasz Wierzbicki, editors, *Mathematical Foundations of Computer Science 1999, 24th International Symposium, MFCS'99*, volume 1672 of *LNCS*, pages 354–365, Szklarska Poręba, Poland, September 6-10 1999. Springer.
6. R. J. van Glabbeek. The linear time-branching time spectrum (Extended abstract). In J. C. M. Baeten and J. W. Klop, editors, *CONCUR '90, Theories of Concurrency: Unification and Extension*, volume 458 of *LNCS*, pages 278–297, Amsterdam, The Netherlands, 27–30 August 1990. Springer-Verlag.
7. Rob van Glabbeek and Ursula Goltz. Equivalence notions for concurrent systems and refinement of actions (Extended abstract). In A. Kreczmar and G. Mirkowska, editors, *Mathematical Foundations of Computer Science 1989*, volume 379 of *LNCS*, pages 237–248, Porąbka-Kozubnik, Poland, August/September 1989. Springer-Verlag.
8. Jan Friso Groote and Hans Hüttel. Undecidable equivalences for basic process algebra. *Information and Computation*, 115(2):354–371, 1994.
9. Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985.
10. Yoram Hirshfeld, Mark Jerrum, and Faron Moller. A polynomial algorithm for deciding bisimilarity of normed context-free processes. *Theoretical Computer Science*, 158(1–2):143–159, 1996.
11. Lalita Jategaonkar and Albert R. Meyer. Deciding true concurrency equivalences on safe, finite nets. *Theoretical Computer Science*, 154:107–143, 1996.
12. André Joyal, Mogens Nielsen, and Glynn Winskel. Bisimulation from open maps. *Information and Computation*, 127(2):164–185, 1996.
13. Paris C. Kanellakis and Scott A. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86(1):43–68, 1990.
14. P. Madhusudan and P. S. Thiagarajan. Controllers for discrete event systems via morphisms. In Davide Sangiorgi and Robert de Simone, editors, *CONCUR'98, Concurrency Theory, 9th International Conference, Proceedings*, volume 1466 of *LNCS*, pages 18–33, Nice, France, September 1998. Springer.
15. R. Milner. *A Calculus of Communicating Systems*, volume 92 of *LNCS*. Springer, 1980.
16. Faron Moller and Scott A. Smolka. On the computational complexity of bisimulation. *ACM Computing Surveys*, 27(2):287–289, 1995.
17. Mogens Nielsen and Christian Clausen. Games and logics for a noninterleaving bisimulation. *Nordic Journal of Computing*, 2(2):221–249, 1995.
18. Mogens Nielsen and Glynn Winskel. Petri nets and bisimulation. *Theoretical Computer Science*, 153(1–2):211–244, 1996.
19. Robert Paige and Robert E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.
20. D. M. R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Theoretical Computer Science: 5th GI-Conference*, volume 104 of *LNCS*, pages 167–183. Springer-Verlag, 1981.
21. A. Rabinovich and B. Trakhtenbrot. Behaviour structures and nets of processes. *Fundamenta Informaticae*, 11:357–404, 1988.

22. Walter Vogler. Deciding history preserving bisimilarity. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo, editors, *Automata, Languages and Programming, 18th International Colloquium, ICALP'91*, volume 510 of *LNCS*, pages 493–505, Madrid, Spain, 8–12 July 1991. Springer-Verlag.
23. Glynn Winskel and Mogens Nielsen. Models for concurrency. In S. Abramsky, Dov M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 4, Semantic Modelling, pages 1–148. Oxford University Press, 1995.