Math. Struct. in Comp. Science (2012), *vol.* 22, *pp.* 175–201.
doi:10.1017/S0960129511000387

Modal logic and the approximation induction principle

MACIEJ GAZDA[†] and WAN FOKKINK[‡]

[†]*Eindhoven University of Technology, Department of Computer Science, P.O.Box 513, 5600 MB Eindhoven, The Netherlands*
Email: m.w.gazda@tue.nl
[‡]*VU University Amsterdam, Department of Computer Science, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands*
Email: wanf@cs.vu.nl

Received 29 January 2010; revised 1 June 2011

We prove a compactness theorem in the context of Hennessy–Milner logic and use it to derive a sufficient condition on modal characterisations for the approximation induction principle to be sound modulo the corresponding process equivalence. We show that this condition is necessary when the equivalence in question is compositional with respect to the projection operators. Furthermore, we derive different upper bounds for the constructive version of the approximation induction principle with respect to simulation and decorated trace semantics.

1. Introduction

Hennessy–Milner logic (Hennessy and Milner 1985) is a modal logic for specifying properties of states in a labelled transition system (LTS). Rob van Glabbeek has used this logic to characterise a wide range of process semantics in terms of observations (van Glabbeek 2001). That is, a process semantics is captured by means of a sublogic of Hennessy–Milner logic so that two states in an LTS are equivalent if and only if they make true exactly the same formulas in this sublogic. In particular, Hennessy–Milner logic itself characterises bisimulation equivalence.

For several process semantics, which are mainly in the realm of simulation, van Glabbeek introduced three different modal characterisations (see van Glabbeek (2001, Figure 9)), which differ in their treatment of conjunction. Apart from the richest characterisations, which correspond to the canonical process equivalences, there are also finitary versions (denoted with a superscript *), which allow only conjunctions over a finite set. Intermediate equivalences based on formulas with arbitrary conjunctions but of finite depth are also considered (with a superscript ω). The corresponding equivalences all differ in general LTSs and collapse in the setting of image-finite LTSs, where an LTS is image-finite if for each state and each action a, there are finitely many outgoing a-transitions. Van Glabbeek sketched separate proofs that the modal characterisations capture the same process semantics under consideration. These proofs are always almost identical.

Here we show that given a modal characterisation of a process semantics for general LTSs, restricting to finite sub-conjunctions produces a modal characterisation of the same semantics for image-finite LTSs. The only requirement is that the formulas obtained in this way were already present in the original modal characterisation. All semantics in the linear time/branching time spectrum (van Glabbeek 2001) have a modal characterisation that satisfies this requirement, except for completed trace semantics (in the case of an infinite action set).

We obtain a similar compactness result for modal characterisations in which formulas have finite depth. In this case, only infinite conjunctions with an infinite depth need to be restricted to their finite sub-conjunctions. Again, the original and resulting semantics coincide on image-finite LTSs if the resulting formulas were already present in the original modal characterisation. The modal characterisation of completed trace semantics satisfies this property.

Van Glabbeek used a version of Hennessy–Milner logic that contains negation (so disjunction, falsum and $[a]\,\phi$ need not be present). However, in that logic the aforementioned result is not so easy to obtain, so we first prove the result in a negation-free version of Hennessy–Milner logic, and then show that the result carries over to Hennessy–Milner logic with negation.

Next we study the approximation induction principle (AIP) from process algebra (Baeten *et al.* 1987), which states that two processes are equal if they are equal up to any finite depth. It is well known that this proof principle is sound modulo bisimulation equivalence for image-finite processes (van Glabbeek 1987). Moreover, it is folklore that this soundness result extends to the other equivalences in the linear time/branching time spectrum (Aceto *et al.* 1994). We obtain a sufficient condition on the modal characterisation of a process equivalence to guarantee that AIP is sound with respect to this equivalence. We then link this result to the compactness theorem from the first part. The sufficient condition says that the modal characterisation must only contain formulas of finite depth. We also show that this is basically a necessary condition: if an equivalence is sound modulo AIP and compositional with respect to the projection operators used in the definition of AIP, then it can be characterised by a set of finite-depth formulas.

Finally, we consider a so-called constructive version of AIP (Mauw 1987), which states that two processes are equal if they are equal up to *some* finite depth. So far, constructive AIP has only been considered for bisimulation equivalence: given an LTS with n states, processes are equal if they are equal at depth $n-1$. Here we provide quadratic bounds for simulation and ready simulation semantics, and, moreover, show that these bounds are sharp. We also provide an exponential bound for decorated trace semantics.

2. Modal characterisations for image-finite processes

2.1. *Hennessy–Milner logic*

A *labelled transition system* (LTS) consists of a set S of states s, a set A of actions a and a set of transitions $s \xrightarrow{a} s'$. An LTS is *image-finite* if for each s and a, the LTS contains only finitely many transitions $s \xrightarrow{a} s'$.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Hennessy–Milner logic (Hennessy and Milner 1985) is a modal logic for specifying properties of states in an LTS. There are several versions of Hennessy–Milner logic. We will write HML to denote the most general language, as presented in van Glabbeek (2001). The syntax of HML can be defined by the following BNF grammar:

$$\varphi \quad ::= \quad \mathsf{T} \mid \bigwedge_{i\in I} \varphi_i \mid \langle a\rangle\varphi \mid \neg\varphi.$$

The meaning of the formulas is defined inductively by

$$\begin{aligned} s &\models \mathsf{T} \\ s &\models \langle a\rangle\varphi \quad \Leftrightarrow \exists s' \in S\,(s \xrightarrow{a} s' \wedge s' \models \varphi) \\ s &\models \textstyle\bigwedge_{i\in I} \varphi_i \Leftrightarrow \forall i \in I\,(s \models \varphi_i) \\ s &\models \neg\varphi \quad \Leftrightarrow s \not\models \varphi. \end{aligned}$$

There is another syntax for Hennessy–Milner logic (see Larsen (1990) and Stirling (2001)), which omits the negation symbol and which we denote by HML^+. As we will see later, its formulas have some nice properties, which make it easier to perform certain proofs. The syntax of HML^+ can be defined by the following BNF grammar:

$$\phi \quad ::= \quad \mathsf{T} \mid \mathsf{F} \mid \bigwedge_{i\in I} \phi_i \mid \bigvee_{i\in I} \phi_i \mid \langle a\rangle\phi \mid [a]\,\phi.$$

The meaning of the new formulas is defined by

$$\begin{aligned} s &\not\models \mathsf{F} \\ s &\models \textstyle\bigvee_{i\in I} \phi_i \Leftrightarrow \exists i \in I\,(s \models \phi_i) \\ s &\models [a]\,\phi \quad \Leftrightarrow \forall s' \in S\,(s \xrightarrow{a} s' \Rightarrow s' \models \phi). \end{aligned}$$

Observe that we allow quantification over arbitrary sets of indexes I. If we restrict to conjunction and disjunction operators over finite sets only, we obtain a language of finite Hennessy–Milner formulas, denoted by HML_{FIN} or HML^+_{FIN}, respectively.

We define the depth of a formula $d : HML \longrightarrow \mathbb{N} \cup \{\infty\}$ inductively by:

$$\begin{aligned} d(\mathsf{T}) &= 0 \\ d\left(\bigwedge_{i\in I} \varphi_i\right) &= \sup\{d(\varphi_i) \mid i \in I\} \\ d(\langle a\rangle\varphi) &= 1 + d(\varphi) \\ d(\neg\varphi) &= d(\varphi) \end{aligned}$$

where $1 + \infty = \infty$. HML_{FDP} and HML^+_{FDP} denote classes of formulas of finite depth:

$$HML^{(+)}_{FDP} = \{\varphi \in HML^{(+)} \mid d(\varphi) < \infty\}.$$

A *context* $C[]$ denotes a formula containing one occurrence of []. The formula $C[\phi]$ is obtained by replacing this occurrence of [] by the formula ϕ. It is well known, and easy to see, that $\phi \Rightarrow \psi$ yields $C[\phi] \Rightarrow C[\psi]$ for all contexts $C[]$ over HML^+ (here $\varphi \Rightarrow \psi$ denotes the fact that for any state s, we have $s \models \varphi \Rightarrow s \models \psi$).

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

2.2. *Compactness results*

In this section we show that for image-finite LTSs, an infinite conjunction or disjunction inside an HML^+ formula can be captured by its finite sub-conjunctions or -disjunctions, respectively. These results are somewhat reminiscent of the compactness theorem for first-order logic, which states that a set of formulas has a model if and only if every finite subset of it has a model.

Lemma 2.8 in Larsen (1990) implies the following proposition, but only for HML^+_{FIN} formulas. Moreover, Larsen (1990) does not provide a proof of Lemma 2.8, so we will include a proof of Proposition 1 to make the current paper self-contained.

$J \subseteq_{\text{FIN}} I$ denotes that J is a finite subset of I.

Proposition 1. Given an image-finite LTS, $s \models C[\bigwedge_{i\in I} \phi_i] \in HML^+$ if and only if $s \models C[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$.

Proof.

($\Rightarrow$)For all $J \subseteq_{\text{FIN}} I$, we have $\bigwedge_{i\in I} \phi_i \Rightarrow \bigwedge_{i\in J} \phi_i$, so $C[\bigwedge_{i\in I} \phi_i] \Rightarrow C[\bigwedge_{i\in J} \phi_i]$.

($\Leftarrow$)We let $s \models C[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$ and apply structural induction on $C[]$ to prove that $s \models C[\bigwedge_{i\in I} \phi_i]$:

— $C[] = []$.
By assumption, $s \models \phi_i$ for all $i \in I$, so $s \models \bigwedge_{i\in I} \phi_i$.

— $C[] = C'[] \wedge \bigwedge_{k\in K} \psi_k$.
$s \models C[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$ implies that $s \models C'[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$, and $s \models \bigwedge_{k\in K} \psi_k$. By induction, the first fact yields $s \models C'[\bigwedge_{i\in I} \phi_i]$. So $s \models C[\bigwedge_{i\in I} \phi_i]$.

— $C[] = C'[] \vee \bigvee_{k\in K} \psi_k$.
If $s \models \psi_{k_0}$ for some $k_0 \in K$, then clearly $s \models C[\bigwedge_{i\in I} \phi_i]$. So we now suppose $s \models C'[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$. Then, by induction, $s \models C'[\bigwedge_{i\in I} \phi_i]$, so $s \models C[\bigwedge_{i\in I} \phi_i]$.

— $C[] = \langle a\rangle C'[]$.
This is the key case.
By assumption, $s \models \langle a\rangle C'[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$. So for each $J \subseteq_{\text{FIN}} I$ there is a state s_J such that $s \xrightarrow{a} s_J$ and $s_J \models C'[\bigwedge_{i\in J} \phi_i]$. Since s is image-finite, $\{s_J \mid J \subseteq_{\text{FIN}} I\}$ is finite, say $\{s_{J_1}, \ldots, s_{J_m}\}$. In order to show a contradiction, we now suppose that $s_{J_k} \not\models C'[\bigwedge_{i\in I} \phi_i]$ for all $k = 1, \ldots, m$. Then, by the induction hypothesis, for all $k = 1, \ldots, m$, we have $s_{J_k} \not\models C'[\bigwedge_{i\in K_k} \phi_i]$ for some $K_k \subseteq_{\text{FIN}} I$. This implies that, for all $k = 1, \ldots, m$, we have $s_{J_k} \not\models C'[\bigwedge_{\ell=1}^{m} \bigwedge_{i\in K_\ell} \phi_i]$. This contradicts the fact that $s_{\bigcup_{\ell=1}^{m} K_\ell} \in \{s_{J_1}, \ldots, s_{J_m}\}$, and we can conclude that $s_{J_{k_0}} \models C'[\bigwedge_{i\in I} \phi_i]$ for some $k_0 \in \{1, \ldots, m\}$. Hence $s \models \langle a\rangle C'[\bigwedge_{i\in I} \phi_i]$.

— $C[] = [a]\, C'[]$.
Let $s \xrightarrow{a} s'$. By assumption, $s' \models C'[\bigwedge_{i\in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$. So, by induction, $s' \models C'[\bigwedge_{i\in I} \phi_i]$. Hence $s \models [a]\, C'[\bigwedge_{i\in I} \phi_i]$. □

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

It is easy to see that Proposition 1 fails for LTSs that are not image-finite. A counterexample is given at the end of Section 2.3, where the top state at the left does not satisfy $\langle a\rangle(\bigwedge_{n\in\mathbb{N}}\langle a\rangle^n\mathsf{T})$, while it does satisfy $\langle a\rangle(\bigwedge_{n\in M}\langle a\rangle^n\mathsf{T})$ for any $M \subseteq_{FIN} \mathbb{N}$.

There is a counterpart of Proposition 1 with disjunction instead of conjunction. To derive this lemma immediately from Proposition 1, we introduce an operator that, given a formula in HML^+, yields a formula equivalent to its negation within HML^+. Given a $\phi \in HML^+$, the formula $\overline{\phi} \in HML^+$ is defined inductively as follows:

$$\overline{\mathsf{T}} = \mathsf{F}$$

$$\overline{\mathsf{F}} = \mathsf{T}$$

$$\overline{\bigwedge_{i\in I}\phi_i} = \bigvee_{i\in I}\overline{\phi_i}$$

$$\overline{\bigvee_{i\in I}\phi_i} = \bigwedge_{i\in I}\overline{\phi_i}$$

$$\overline{\langle a\rangle\phi} = [a]\,\overline{\phi}$$

$$\overline{[a]\,\phi} = \langle a\rangle\overline{\phi}.$$

Clearly, $\neg\phi \Leftrightarrow \overline{\phi}$. Moreover, $\overline{\overline{\phi}} = \phi$. The definition is extended to contexts by putting $\overline{[]} = []$. We write $\overline{C}[]$ for $\overline{C[]}$. It is easy to see that $\overline{C[\phi]} = \overline{C}[\overline{\phi}]$.

Proposition 2. Given an image-finite LTS, $s \models C[\bigvee_{i\in I}\phi_i]$ if and only if $s \models C[\bigvee_{i\in J}\phi_i]$ for some $J \subseteq_{\mathrm{FIN}} I$.

Proof.

$$\begin{aligned}
s \models C\left[\bigvee_{i\in I}\phi_i\right] &\Leftrightarrow s \not\models \overline{C\left[\bigvee_{i\in I}\phi_i\right]} \\
&\Leftrightarrow s \not\models \overline{C}\left[\bigwedge_{i\in I}\overline{\phi_i}\right] \\
&\Leftrightarrow s \not\models \overline{C}\left[\bigwedge_{i\in J}\overline{\phi_i}\right] \text{ for some } J \subseteq_{\mathrm{FIN}} I \qquad \text{(by Proposition 1)} \\
&\Leftrightarrow s \not\models \overline{C\left[\bigvee_{i\in J}\phi_i\right]} \text{ for some } J \subseteq_{\mathrm{FIN}} I \\
&\Leftrightarrow s \models C\left[\bigvee_{i\in J}\phi_i\right] \text{ for some } J \subseteq_{\mathrm{FIN}} I. \qquad \square
\end{aligned}$$

We now consider Hennessy–Milner logic with negation, *viz. HML*. Contexts over this syntax are denoted by $D[]$. Each formula φ over this logic can be translated to an

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

equivalent formula $P(\varphi) \in HML^+$ in a straightforward fashion:

$$P(\mathsf{T}) = \mathsf{T}$$
$$P(\langle a \rangle \varphi) = \langle a \rangle P(\varphi)$$
$$P\left(\bigwedge_{i \in I} \varphi_i\right) = \bigwedge_{i \in I} P(\varphi_i)$$
$$P(\neg \varphi) = \overline{P(\varphi)}.$$

Clearly, $\varphi \Leftrightarrow P(\varphi)$. The definition is extended to contexts by putting $P([]) = []$. We write $P(D)[]$ for $P(D[])$.

For Hennessy–Milner logic with negation, we define *positive* and *negative* contexts inductively as follows:

— $[]$ is a positive context.
— If $D[]$ is a positive (respectively, negative) context, then $D[] \wedge \bigwedge_{i \in I} \varphi_i$ and $\langle a \rangle D[]$ are positive (respectively, negative) contexts.
— If $D[]$ is a positive (respectively, negative) context, then $\neg D[]$ is a negative (respectively, positive) context.

Lemma 1.

— If $D[]$ is a positive context, $P(D[\varphi]) = P(D)[P(\varphi)]$.
— If $D[]$ is a negative context, $P(D[\varphi]) = P(D)[\overline{P(\varphi)}]$.

Proof. We prove both statements simultaneously, by structural induction on $D[]$. The cases where $D[]$ is of the form $[]$, $D'[] \wedge \bigwedge_{i \in I} \varphi_i$ or $\langle a \rangle D'[]$ are straightforward and left as an exercise. We focus on the key case, where $D[] = \neg D'[]$:

First let $D[]$ be positive, so $D'[]$ is negative. Then

$$\begin{aligned} P(\neg D'[\varphi]) &= \overline{P(D'[\varphi])} \\ &= \overline{P(D')[\overline{P(\varphi)}]} && \text{(by induction)} \\ &= \overline{P(D')}[\overline{\overline{P(\varphi)}}] \\ &= P(\neg D')[P(\varphi)]. \end{aligned}$$

Next let $D[]$ be negative, so $D'[]$ is positive. Then

$$\begin{aligned} P(\neg D'[\varphi]) &= \overline{P(D'[\varphi])} \\ &= \overline{P(D')[P(\varphi)]} && \text{(by induction)} \\ &= \overline{P(D')}[\overline{P(\varphi)}] \\ &= P(\neg D')[\overline{P(\varphi)}]. \end{aligned}$$

□

We can now prove a counterpart of Propositions 1 and 2 for *HML*.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Proposition 3. Given an image-finite LTS.

(1) If $D[]$ is a positive context, then $s \models D[\bigwedge_{i\in I} \varphi_i]$ if and only if $s \models D[\bigwedge_{i\in J} \varphi_i]$ for all $J \subseteq_{\mathrm{FIN}} I$.

(2) If $D[]$ is a negative context, then $s \models D[\bigwedge_{i\in I} \varphi_i]$ if and only if $s \models D[\bigwedge_{i\in J} \varphi_i]$ for some $J \subseteq_{\mathrm{FIN}} I$.

Proof.

(1)If $D[]$ is a positive context,

$$
\begin{aligned}
s \models D\left[\bigwedge_{i\in I} \varphi_i\right] &\Leftrightarrow s \models P\left(D\left[\bigwedge_{i\in I} \varphi_i\right]\right) \\
&\Leftrightarrow s \models P(D)\left[\bigwedge_{i\in I} P(\varphi_i)\right] && \text{(by Lemma 1)} \\
&\Leftrightarrow s \models P(D)\left[\bigwedge_{i\in J} P(\varphi_i)\right] \text{ for all } J \subseteq_{\mathrm{FIN}} I && \text{(by Proposition 1)} \\
&\Leftrightarrow s \models P\left(D\left[\bigwedge_{i\in J} \varphi_i\right]\right) \text{ for all } J \subseteq_{\mathrm{FIN}} I && \text{(by Lemma 1)} \\
&\Leftrightarrow s \models D\left[\bigwedge_{i\in J} \varphi_i\right] \text{ for all } J \subseteq_{\mathrm{FIN}} I.
\end{aligned}
$$

(2)If $D[]$ is a negative context,

$$
\begin{aligned}
s \models D\left[\bigwedge_{i\in I} \varphi_i\right] &\Leftrightarrow s \models P\left(D\left[\bigwedge_{i\in I} \varphi_i\right]\right) \\
&\Leftrightarrow s \models P(D)\left[\bigvee_{i\in I} \overline{P(\varphi_i)}\right] && \text{(by Lemma 1)} \\
&\Leftrightarrow s \models P(D)\left[\bigvee_{i\in J} \overline{P(\varphi_i)}\right] \text{ for some } J \subseteq_{\mathrm{FIN}} I && \text{(by Proposition 2)} \\
&\Leftrightarrow s \models P\left(D\left[\bigwedge_{i\in J} \varphi_i\right]\right) \text{ for some } J \subseteq_{\mathrm{FIN}} I && \text{(by Lemma 1)} \\
&\Leftrightarrow s \models D\left[\bigwedge_{i\in J} \varphi_i\right] \text{ for some } J \subseteq_{\mathrm{FIN}} I.
\end{aligned}
$$

□

2.3. *Modal characterisations*

A process semantics on LTSs can be captured by means of a sublogic of *HML* – see Bloom *et al.* (2004) and van Glabbeek (2001) for a wide range of such modal characterisations. Given such a sublogic $\mathcal{O}$, two states in an LTS are equivalent if and only if they make exactly the same formulas in $\mathcal{O}$ true. We denote this equivalence relation on states by $\sim_{\mathcal{O}}$.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

We prove that given such a modal characterisation of a process semantics for general LTSs, restricting infinite conjunctions to their finite sub-conjunctions produces a modal characterisation of the same semantics on image-finite LTSs. The only requirement is that these finite sub-conjunctions are already present in the original modal characterisation for general LTSs.

We obtain a similar compactness result for modal characterisations for which the formulas may contain infinite conjunctions, but are all of finite depth. In this case only infinite conjunctions that have an infinite depth need to be restricted to their finite sub-conjunctions. Again, both the original and resulting semantics coincide if the resulting formulas were already present in the original modal characterisation.

The modal characterisations in Bloom *et al.* (2004) all satisfy this requirement, except for the one for completed trace semantics in the case of an infinite action set. Specifically, the modal characterisation of completed trace semantics for general LTSs, as well as for image-finite ones, is

$$\varphi \ ::= \ \mathsf{T} \ \mid \ \bigwedge_{a\in A} \neg\langle a\rangle\mathsf{T} \ \mid \ \langle a\rangle\varphi$$

where A denotes the set of all actions.

Given a modal characterisation $\mathscr{O}$, we use $\mathscr{O}_{\mathrm{FIN}}$ to denote the sublogic of formulas in $\mathscr{O}$ that do not contain infinite conjunctions, and $\mathscr{O}_{FDP}$ for the sublogic of formulas with finite depth. Clearly, $\mathscr{O}_{\mathrm{FIN}} \subseteq \mathscr{O}_{FDP}$. Using the results from Section 2.2, we can now prove the main theorem of this section.

Theorem 1. Given an image-finite LTS and $\mathscr{O} \subseteq HML$.

(1) If for each $D[\bigwedge_{i\in I}\varphi_i] \in \mathscr{O}$ with I infinite and $d(\bigwedge_{i\in I}\varphi_i) = \infty$ we have $D[\bigwedge_{i\in J}\varphi_i] \in \mathscr{O}$ for all $J \subseteq_{\mathrm{FIN}} I$, then $\sim_{\mathscr{O}}$ and $\sim_{\mathscr{O}_{\mathrm{FDP}}}$ coincide.
(2) If for each $D[\bigwedge_{i\in I}\varphi_i] \in \mathscr{O}$ with I infinite we have $D[\bigwedge_{i\in J}\varphi_i] \in \mathscr{O}$ for all $J \subseteq_{\mathrm{FIN}} I$, then $\sim_{\mathscr{O}}$ and $\sim_{\mathscr{O}_{\mathrm{FIN}}}$ coincide.

Proof. We will prove the theorem for the subset of finite formulas $\mathscr{O}_{\mathrm{FIN}}$, and provide comments in square brackets for the version with $\mathscr{O}_{\mathrm{FDP}}$ where appropriate. Since $\mathscr{O}_{\mathrm{FIN}} \subseteq \mathscr{O}_{\mathrm{FDP}} \subseteq \mathscr{O}$, we clearly have $\sim_{\mathscr{O}} \subseteq \sim_{\mathscr{O}_{\mathrm{FDP}}} \subseteq \sim_{\mathscr{O}_{\mathrm{FIN}}}$. We need to show that $\mathscr{O}_{\mathrm{FIN}}$ [respectively, $\mathscr{O}_{\mathrm{FDP}}$] can distinguish all the states that $\mathscr{O}$ can.

Given states s, s' and a formula $\varphi \in \mathscr{O}$ with $s \models \varphi$ and $s' \not\models \varphi$. We construct a formula in $\mathscr{O}_{\mathrm{FIN}}$ [respectively, $\mathscr{O}_{\mathrm{FDP}}$] that distinguishes s and s'. We apply ordinal induction on the height $\lambda(\varphi)$ of the tree of infinite conjunctions [of infinite depth] in φ. That is,

$$\begin{aligned}
\lambda(\mathsf{T}) &= 0\\
\lambda(\langle a\rangle\varphi) &= \lambda(\varphi)\\
\lambda(\textstyle\bigwedge_{i\in I}\varphi_i) &= \begin{cases} \sup\{\lambda(\varphi_i) \mid i\in I\}+1 & \text{if } I \text{ is infinite [and } d(\bigwedge_{i\in I}\varphi_i)=\infty]\\ \sup\{\lambda(\varphi_i) \mid i\in I\} & \text{otherwise}\end{cases}\\
\lambda(\neg\varphi) &= \lambda(\varphi).
\end{aligned}$$

The base case is trivial since if $\lambda(\varphi) = 0$, then $\varphi \in \mathscr{O}_{\mathrm{FIN}}$ [respectively, $\varphi \in \mathscr{O}_{\mathrm{FDP}}$].

We now consider the inductive case, where $\lambda(\varphi) > 0$. Let $\varphi = D[\bigwedge_{i\in I}\varphi_i]$ with I [and $d(\bigwedge_{i\in I}\varphi_i)$] infinite, where this occurrence of an infinite conjunction [and depth] in φ is

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

outermost in the sense that it does not occur within any infinite conjunction [of infinite depth]. We distinguish two cases:

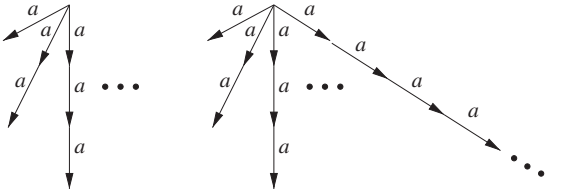
— $D[]$ is a positive context.
  By Proposition 3(1), $s' \not\models \varphi$ implies that $s' \not\models D[\bigwedge_{i\in J_0} \varphi_i]$ for some $J_0 \subseteq_{\mathrm{FIN}} I$, while $s \models \varphi$ implies that $s \models D[\bigwedge_{i\in J_0} \varphi_i]$.
— $D[]$ is a negative context.
  By Proposition 3(2), $s \models \varphi$ implies that $s \models D[\bigwedge_{i\in J_0} \varphi_i]$ for some $J_0 \subseteq_{\mathrm{FIN}} I$, while $s' \not\models \varphi$ implies that $s' \not\models D[\bigwedge_{i\in J_0} \varphi_i]$.

In both cases, $D[\bigwedge_{i\in J_0} \varphi_i] \in \mathcal{O}$ by assumption.

It is clear that there are only finitely many outermost occurrences of infinite conjunctions [of infinite depth] in φ. Using the construction above, these can all be replaced by finite conjunctions to obtain a formula $\psi \in \mathcal{O}$ that distinguishes s and s'. Since $\lambda(\psi) < \lambda(\varphi)$, we can use ordinal induction to construct a formula in $\mathcal{O}_{\mathrm{FIN}}$ [respectively, $\mathcal{O}_{\mathrm{FDP}}$] that distinguishes s and s'. □

It is easy to see that the requirement in Theorem 1 that $D[\bigwedge_{i\in J} \varphi_i] \in \mathcal{O}$ for all $J \subseteq_{\mathrm{FIN}} I$ cannot be omitted. For instance, let $\mathcal{O}$ consist of a single formula with an infinite conjunction, $\bigwedge_{n\in\mathbb{N}} \langle a\rangle^n \mathsf{T}$ (with $\langle a\rangle^0 \varphi = \varphi$ and $\langle a\rangle^{n+1}\varphi = \langle a\rangle(\langle a\rangle^n \varphi)$). Then $\mathcal{O}_{\mathrm{FIN}} = \varnothing$, so $\sim_{\mathcal{O}_{\mathrm{FIN}}}$ is the universal relation. On the other hand, $\mathcal{O}$ distinguishes an a-cycle from a deadlock state.

The following example, which is taken from van Glabbeek (1987), shows that Theorem 1 fails for LTSs that are not image-finite. Consider an LTS that consists of finite a-traces of arbitrary length, and an LTS that on top of this exhibits an infinite a-trace.



Let

$$\mathcal{O} = \{\langle a\rangle \left(\bigwedge_{n\in N} \langle a\rangle^n \mathsf{T} \right) \mid N \subseteq \mathbb{N}\}.$$

Then

$$\mathcal{O}_{\mathrm{FIN}} = \{\langle a\rangle \left(\bigwedge_{n\in N} \langle a\rangle^n \mathsf{T} \right) \mid N \subseteq_{\mathrm{FIN}} \mathbb{N}\}.$$

Clearly, $\mathcal{O}$ distinguishes the top states of the two LTSs above by means of any formula

$$\langle a\rangle \left(\bigwedge_{n\in N} \langle a\rangle^n \mathsf{T} \right)$$

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

with N infinite. Namely, such a formula holds for the top state at the right, but not for the top state at the left. However, $\mathcal{O}_{\mathrm{FIN}}$ does not distinguish these states since all formulas in $\mathcal{O}_{\mathrm{FIN}}$ hold for both states.

Goldblatt (1995) and Hollenberg (1995) (see also Blackburn *et al.* (2001)) investigated models that are more general than image-finite LTSs, but do have the Hennessy–Milner property. That is, models where the modal equivalence $\sim_{HML}$ coincides with bisimulation equivalence. This led to the notion of modally saturated processes – an LTS is M-saturated if for all states s and all $\mathcal{O} \subseteq HML$, whenever every finite subset of $\mathcal{O}$ is satisfied in some a-successor of s, there exists an a-successor of s in which $\mathcal{O}$ is satisfied. It is not difficult to prove using ordinal induction on the structure of formulas that Theorem 1 also holds for M-saturated models.

3. The approximation induction principle

For each natural number n, we define a *projection operator* π_n that mimics the behaviour of its argument up to n steps and then terminates. The behaviour of an application of the projection operator to a process (or state) is given by the following rule scheme:

$$\frac{x \xrightarrow{a} x'}{\pi_{n+1}(x) \xrightarrow{a} \pi_n(x')}.$$

The *Approximation Induction Principle (AIP)* states that if two processes are equal up to any finite depth, then the processes themselves are equal.

(AIP) If $\pi_n(x) = \pi_n(y)$ for all $n \in N$, then $x = y$.

3.1. *Sufficient criterion for soundness of AIP*

Aceto *et al.* (1994) states that AIP is sound for all eleven 'strong' equivalences found in van Glabbeek (2001), but does not give any argument in support of this assertion. The soundness of AIP has been proved several times for bisimulation equivalence (for example, van Glabbeek (1987)) in the setting of finitely branching or image-finite LTSs. The standard technique is to prove that a relation identifying two processes if and only if all of their projections are bisimilar is a bisimulation (provided one of the processes is image-finite). A different proof was presented in Baeten and Weijland (1990), where, given two processes p and q, they consider, for all $n \in \mathbb{N}$, the bisimulations between $\pi_n(p)$ and $\pi_n(q)$. Bisimulations for the nth projection are linked with those bisimulations for the $(n+1)$th projection in which they are included. In this way, an infinite, finitely branching tree is constructed. The bisimulation between p and q is a sum of bisimulations lying on an infinite path in the tree.

We present a general proof of soundness of AIP in a different way for a range of equivalences using properties of modal languages that define an equivalence. Specifically, AIP is sound for all process equivalences that can be defined using modal characterisations within HML_{FDP}. The crucial part of the proof is the following lemma, which states that if a finite-depth formula is satisfied by a process, then it is satisfied by almost all of its projections.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Lemma 2. Given any LTS, for all states s and $\varphi \in HML_{FDP}$, we have

$$s \models \varphi \Leftrightarrow \forall n \geqslant d(\varphi)\,(\pi_n(s) \models \varphi).$$

Proof. We use ordinal induction on the complexity of a formula, defined by

$$\begin{aligned} |\mathsf{T}| &= 1 \\ |\langle a\rangle\varphi| &= |\varphi| + 1 \\ |\bigwedge_{i\in I} \varphi_i| &= \sup\{|\varphi_i| \mid i \in I\} + 1 \\ |\neg\varphi| &= |\varphi| + 1. \end{aligned}$$

($\Rightarrow$) The base is trivial ($\varphi = \mathsf{T}$).
Let s be an arbitrary state and φ be a formula such that $s \models \varphi$, and suppose that for all s' and all ψ with $|\psi| < |\varphi|$, we have $s' \models \psi$ implies that ψ is satisfied by all projections $\pi_n(s')$ for $n \geqslant d(\psi)$. There are three possible cases:

— $\varphi = \langle a\rangle\psi$
Then there exists a q with $s \xrightarrow{a} q$ and $q \models \psi$. From the induction hypothesis, we obtain

$$\forall n \geqslant d(\psi)\,(\pi_n(q) \models \psi).$$

Since

$$\pi_n(s) \xrightarrow{a} \pi_{n-1}(q)$$

for $n \geqslant 1$, we have

$$\forall n \geqslant d(\psi) + 1\,(\pi_n(s) \models \langle a\rangle\psi),$$

so

$$\forall n \geqslant d(\langle a\rangle\psi)\,(\pi_n(s) \models \langle a\rangle\psi)$$

— $\varphi = \bigwedge_{i\in I} \psi_i$
So $\forall i \in I\ (s \models \psi_i)$. By induction, this implies

$$\forall i \in I\ \forall n \geqslant d(\psi_i)\,(\pi_n(s) \models \psi_i).$$

Therefore,

$$\forall n \geqslant max_{i\in I}\{d(\psi_i)\}\ \forall i \in I\ (\pi_n(s) \models \psi_i).$$

By definition

$$d\left(\bigwedge_{i\in I} \psi_i\right) = max_{i\in I}\{d(\psi_i)\},$$

so

$$\forall n \geqslant d\left(\bigwedge_{i\in I} \psi_i\right)\ \left(\pi_n(s) \models \bigwedge_{i\in I} \psi_i\right).$$

— $\varphi = \neg\psi$
We have to consider all the subcases, depending on ψ:

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

- $\psi = \mathsf{T}$:
  This case is impossible since it would mean that $s \not\models \mathsf{T}$, which is never true.
- $\psi = \langle a \rangle \psi'$:
  So for all s' with $s \xrightarrow{a} s'$, we have $s' \models \neg\psi'$, and by induction
  $$\forall n \geqslant d(\neg\psi')\,(\pi_n(s') \models \neg\psi').$$
  Therefore
  $$\forall n \geqslant d(\neg\psi') + 1\,(\pi_n(s) \models \neg\langle a \rangle \psi'),$$
  and thus
  $$\forall n \geqslant d(\psi)\,(\pi_n(s) \models \neg\langle a \rangle \psi').$$
- $\psi = \bigwedge_{i \in I} \psi_i$:
  So $\exists i_0 \in I\,(s \models \neg\psi_{i_0})$. By induction,
  $$\forall n \geqslant d(\neg\psi_{i_0})\,(\pi_n(s) \models \neg\psi_{i_0}),$$
  so
  $$\forall n \geqslant d(\varphi)\,(\pi_n(s) \models \neg \bigwedge_{i \in I} \psi_i),$$
  which is the desired result.
- $\psi = \neg\psi'$:
  This case is immediate since φ is equivalent to ψ'.

($\Leftarrow$) This direction follows immediately from what we have just proved. To see this, take an arbitrary formula $\varphi \in \mathcal{O}$ and a state s such that $\forall n \geqslant d(\varphi)\,(\pi_n(s) \models \varphi)$. In order to show a contradiction, we now suppose that $s \not\models \varphi$. This means $s \models \neg\varphi$, but we have already proved that this implies $\forall n \geqslant d(\neg\varphi)\,(\pi_n(s) \models \neg\varphi)$, which contradicts our assumptions. Therefore s must satisfy φ. □

Theorem 2. If $\mathcal{O} \subseteq HML_{FDP}$, we have AIP is sound for $\sim_{\mathcal{O}}$.

Proof. We need to show that

$$\forall n \in N\,(\pi_n(s) \sim_{\mathcal{O}} \pi_n(q)) \Rightarrow s \sim_{\mathcal{O}} q.$$

Suppose

$$\forall n \in N\,(\pi_n(s) \sim_{\mathcal{O}} \pi_n(q)).$$

We have to prove that $\mathcal{O}(s) = \mathcal{O}(q)$. In fact, it suffices to prove that $\mathcal{O}(s) \subseteq \mathcal{O}(q)$ since the proof of the other inclusion is symmetric. Take any $\varphi \in \mathcal{O}(s)$. According to Lemma 2, we have

$$\forall n \geqslant d(\varphi)\,(\varphi \in \mathcal{O}(\pi_n(s)) = \mathcal{O}(\pi_n(q))),$$

and using the same lemma again, we obtain $\varphi \in \mathcal{O}(q)$. □

Using the results from the previous section, we can now obtain the following sufficient condition for the soundness of AIP in the setting of image-finite LTSs.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Corollary 1. Let $\mathcal{O} \subseteq HML$. Suppose that for each $D[\bigwedge_{i\in I} \varphi_i] \in \mathcal{O}$ with I infinite and $d(\bigwedge_{i\in I} \varphi_i) = \infty$, we have $D[\bigwedge_{i\in J} \varphi_i] \in \mathcal{O}$ for all $J \subseteq_{\mathrm{FIN}} I$. Then AIP is sound for $\sim_{\mathcal{O}}$ in the setting of image-finite LTSs.

Proof. If $\mathcal{O}$ meets the above requirements, then according to Theorem 1 (2), we have $\sim_{\mathcal{O}} = \sim_{\mathcal{O}'}$, where $\mathcal{O}' \in HML_{FDP}$. Hence, by Theorem 2, AIP is sound for $\sim_{\mathcal{O}}$. □

Corollary 2. AIP is sound with respect to all the basic process equivalences on image-finite LTSs, namely trace, completed trace, failures, readiness, failure trace, ready trace, ready simulation, n-nested simulation ($n \geqslant 1$) and bisimulation.

Proof. As pointed out in van Glabbeek (2001), all the above equivalences with the exception of completed trace can be defined with a sublogic of Hennessy–Milner logic consisting only of finite formulas. Moreover, all formulas in the modal language corresponding to completed trace equivalence are finite-depth. □

3.2. *Necessary criterion for soundness of AIP*

Soundness of AIP does not necessarily imply that the equivalence in question is definable with a sublogic of HML_{FDP}. Observe first that having a fixed set of actions A, for any formula $\varphi \in HML$, we can express an ACTL formula $E\varphi$ ('there exists an execution path to a state in which φ holds') using a single formula from HML. Indeed, for any $\varphi \in HML$, the formula $\bigvee_{\sigma\in A^*} \sigma\varphi$ is equivalent to $E\varphi$. Now consider an equivalence relating two processes according to whether action a can be executed in at least one execution path (that is, if $E(\langle a\rangle \mathsf{T})$ is satisfied). It is easy to see that AIP is sound for this equivalence, but it cannot be defined with a sublogic of HML_{FDP}.

In this section we consider only those equivalences that are compositional with respect to projection operators (this includes all the equivalences mentioned in Corollary 2). We prove that in this class, definability of an equivalence with finite-depth formulas is also a necessary condition for the soundness of AIP.

First we define for each $\varphi \in HML$ a corresponding formula $cut_n(\varphi) \in HML_{FDP}$ in which every subformula of the form $\langle a\rangle\psi$ appearing at depth n is replaced by F. The functions $cut_n : HML \to HML_{FDP}$ for $n \in \mathbb{N}$ are defined inductively as follows:

$$
\begin{aligned}
cut_n(\mathsf{T}) &= \mathsf{T} \\
cut_0(\langle a\rangle\varphi) &= \mathsf{F} \\
cut_n(\neg\varphi) &= \neg cut_n(\varphi) \\
cut_n\left(\bigwedge_{i\in I} \varphi_i\right) &= \bigwedge_{i\in I} cut_n(\varphi_i) \\
cut_{n+1}(\langle a\rangle\varphi) &= \langle a\rangle cut_n(\varphi)
\end{aligned}
$$

We now prove a key property for *cut* functions.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Lemma 3. Given any LTS, for all states s, $\varphi \in HML$ and $n \in \mathbb{N}$, we have

$$\pi_n(s) \models \varphi \Leftrightarrow s \models cut_n(\varphi) \tag{CT}$$

Proof. We prove CT by induction on the structure of φ:

— $\varphi = \mathsf{T}$:
$\pi_n(s) \models \mathsf{T}$ and $s \models cut_n(\mathsf{T}) = \mathsf{T}$.

— $\varphi = \langle a \rangle \psi$:
We distinguish cases where $n = 0$ and $n > 0$.

– $n = 0$:
Clearly, $\pi_0(s) \not\models \langle a \rangle \psi$ and $s \not\models cut_0(\langle a \rangle \psi) = \mathsf{F}$.

– $n > 0$:
We have

$$\begin{aligned}
\pi_n(s) \models \langle a \rangle \psi &\Leftrightarrow \exists s' \, (s \xrightarrow{a} s' \wedge \pi_{n-1}(s') \models \psi) && \text{(transition rules for } \pi_{n-1}) \\
&\Leftrightarrow \exists s' \, (s \xrightarrow{a} s' \wedge \; s' \models cut_{n-1}(\psi)) && \text{(structural induction)} \\
&\Leftrightarrow s \models \langle a \rangle cut_{n-1}(\psi) \\
&\Leftrightarrow s \models cut_n(\langle a \rangle \psi). && \text{(definition of } cut)
\end{aligned}$$

— $\varphi = \bigwedge_{i \in I} \psi_i$:

$$\begin{aligned}
\pi_n(s) \models \bigwedge_{i \in I} \psi_i &\Leftrightarrow \forall i \in I \, (\pi_n(s) \models \psi_i) \\
&\Leftrightarrow \forall i \in I \, (s \models cut_n(\psi_i)) && \text{(structural induction)} \\
&\Leftrightarrow s \models cut_n \left(\bigwedge_{i \in I} \psi_i \right). && \text{(definition of } cut)
\end{aligned}$$

— $\varphi = \neg \psi$:

$$\begin{aligned}
\pi_n(s) \models \neg \psi &\Leftrightarrow \pi_n(s) \not\models \psi \\
&\Leftrightarrow s \not\models cut_n(\psi) && \text{(structural induction)} \\
&\Leftrightarrow s \models \neg cut_n(\psi) \\
&\Leftrightarrow s \models cut_n(\neg \psi). && \text{(definition of } cut)
\end{aligned}$$

□

Theorem 3. Suppose $\sim_{\mathcal{O}}$ is a process equivalence induced by some $\mathcal{O} \subseteq HML$ and compositional with respect to all projection operators π_n. AIP is sound for $\sim_{\mathcal{O}}$ if and only if $\sim_{\mathcal{O}}$ can be defined with some $\mathcal{O}_1 \subseteq HML_{FDP}$.

Proof.

($\Leftarrow$) The fact that definability of an equivalence with a sublogic of HML_{FDP} implies the soundness of AIP was proved in Theorem 2.

($\Rightarrow$) We have to prove that the soundness of AIP implies

$$\exists \mathcal{O}_1 \subseteq HML_{FDP} \, (s \sim_{\mathcal{O}} q \Leftrightarrow s \sim_{\mathcal{O}_1} q).$$

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

The desired $\mathcal{O}_1$ is constructed by applying the cut_n functions to formulas from $\mathcal{O}$:

$$\mathcal{O}_1 = \bigcup_{n\in\mathbb{N}} \{cut_n(\varphi) \mid \varphi \in \mathcal{O}\}.$$

We have:

$$\begin{aligned}
s \sim_{\mathcal{O}} q &\Leftrightarrow \forall n \in \mathbb{N}\,(\pi_n(s) \sim_{\mathcal{O}} \pi_n(q)) && \text{(soundness of AIP for } \sim_{\mathcal{O}} \text{ and)}\\
&&& \text{(compositionality with respect to projection)}\\
&\Leftrightarrow \forall n \in \mathbb{N}\,\forall \varphi \in \mathcal{O}\,(\pi_n(s) \models \varphi \Leftrightarrow \pi_n(q) \models \varphi)\\
&\Leftrightarrow \forall n \in \mathbb{N}\,\forall \varphi \in \mathcal{O}\,(s \models cut_n(\varphi) \Leftrightarrow q \models cut_n(\varphi)) && \text{(Lemma 3)}\\
&\Leftrightarrow \forall n \in \mathbb{N}\,\forall \psi \in \mathcal{O}_1\,(s \models \psi \Leftrightarrow q \models \psi) && \text{(definition of } \mathcal{O}_1)\\
&\Leftrightarrow \forall n \in \mathbb{N}\,(\pi_n(s) \sim_{\mathcal{O}_1} \pi_n(q))\\
&\Leftrightarrow s \sim_{\mathcal{O}_1} q.
\end{aligned}$$

□

4. Constructive AIP

The approximation induction principle is not very attractive in the computational sense since it requires us to check infinitely many projections of two processes to derive their equivalence. To overcome this drawback, a notion of constructive AIP (AIP^c) was introduced in Mauw (1987) and investigated further in Barros and Hou (2008). AIP^c makes an additional assumption that LTSs are regular (that is, have a finite number of states and transitions). Such LTSs can be defined using n linear equations for some $n \in \mathbb{N}$; each equation defines a state and its outgoing transitions. AIP^c specifies a value $K \in N$ for which equivalence between two states can be decided by comparing their Kth projections. Constructive AIP has so far only been considered for bisimulation equivalence, for which it has been proved that if $\pi_{n-1}(p) \underline{\leftrightarrow} \pi_{n-1}(q)$, then $p \underline{\leftrightarrow} q$, where n is the number of linear equations defining an LTS that contains p and q (Barros and Hou 2008).

We investigate the possibility of obtaining similar results for other process semantics. In this section our approach is different in two ways: we focus separately on some particular semantics, and also consider AIP^c from the preorder perspective. The latter approach provides more general statements, and as we focus especially on simulation-based semantics, the most natural way to obtain results for process equivalences is to tackle their preorder counterparts first.

From now on, we will only consider regular LTSs. For a preorder $\sqsubseteq$, we use $D(\sqsubseteq, N)$ to denote the smallest number such that for all regular LTSs with N distinct states,

$$\pi_{D(\sqsubseteq,N)}(p) \sqsubseteq \pi_{D(\sqsubseteq,N)}(q) \;\Rightarrow\; p \sqsubseteq q,$$

and we introduce a similar notation $D(\sim, N)$ for equivalences.

We assume a preorder $(P, \preceq)$ and use $\prec$ to denote the corresponding strong order relation, that is, $x \prec y$ if $x \preceq y$ and $x \neq y$. We say that b is *directly above* a if and only if $a \prec b$ and there is no c such that $a \prec c \prec b$. We further say that a is $\preceq$-*equivalent* to b if $a \preceq b$ and $b \preceq a$. We use Δ to denote an identity relation.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

4.1. *Simulation-based semantics*

We recall the definition of simulation and ready simulation semantics. We use $I(p)$ to denote $\{a \in A \mid \exists p' (p \xrightarrow{a} p')\}$. A relation $R \in P \times P$ is a [ready] simulation relation if:

$pRq \Rightarrow$ for all p' with $p \xrightarrow{a} p'$ there exists a $q \in P$ such that $q \xrightarrow{a} q'$ and $p'Rq'$ [and $I(p) = I(q)$].

We define the [ready] simulation preorder by:

$p \sqsubseteq^S q$ [respectively, $p \sqsubseteq^{RS} q$] if there exists a [ready] simulation relation R such that pRq.

For $X \in \{S, RS\}$ and $K \geqslant 0$, we use $\sqsubseteq^X_K$ to denote the [ready] simulation preorder up to K steps:

$$p \sqsubseteq^X_K q \text{ if and only if } \pi_K(p) \sqsubseteq^X \pi_K(q).$$

Lemma 4. For $X \in \{S, RS\}$, if $\sqsubseteq^X_{K+1} = \sqsubseteq^X_K$ for some $K \in \mathbb{N}$, then $\sqsubseteq^X_N = \sqsubseteq^X_K$ for all $N > K$.

Proof. We present a proof for simulation semantics with additional remarks for ready simulation. In order to show a contradiction, we suppose that there exists an $N > K + 1$ such that $\sqsubseteq^S_N \neq \sqsubseteq^S_K$. Let N be the lowest number with this property. Since it is clear that $\sqsubseteq^S_N \subseteq \sqsubseteq^S_{N-1}$, there must exist processes p, q such that $p \sqsubseteq^S_{N-1} q$, but $p \not\sqsubseteq^S_N q$. So there exists a transition $p \xrightarrow{a} p'$ that cannot be mimicked by q, that is, $\neg\exists q' (q \xrightarrow{a} q' \wedge p' \sqsubseteq^S_{N-1} q')$ [in the ready simulation case, we have to consider the case when $I(p) \neq I(q)$, but this is impossible since $p \sqsubseteq^{RS}_{N-1} q$ and $N - 1 \geqslant 1$]. Since $p \sqsubseteq^S_{N-1} q$, we know that $\exists q' (q \xrightarrow{a} q' \wedge p' \sqsubseteq^S_{N-2} q')$ and since $\sqsubseteq^S_{N-1} = \sqsubseteq^S_{N-2}$, we have $p' \sqsubseteq^S_{N-1} q'$, which gives a contradiction. □

Since

— $\sqsubseteq^S \subseteq \sqsubseteq^S_{K+1} \subseteq \sqsubseteq^S_K$ for all $K \in \mathbb{N}$,
— $\sqsubseteq^S_0$ relates all N^2 possible pairs, and
— $\sqsubseteq^S$ includes the N identity pairs,

we obtain the following corollary.

Corollary 3. $D(\sqsubseteq^S, N) \leqslant N^2 - N$.

The above upper bound estimation is asymptotically the same as the actual value of $D(\sqsubseteq^S, N)$, which will be shown to be

$$(N + 2)(N - 1)/2.$$

To explain the ideas underlying AIP^c for the (ready) simulation preorder, we first prove that

$$D(\sqsubseteq^S, N) \leqslant \frac{(N + 2)(N - 1)}{2}.$$

This is an immediate consequence of a more general preorder property, namely, that for any preorder $(P, \sqsubseteq)$ with $|P| = N$, we can refine $\sqsubseteq$ in at most

$$(N + 2)(N - 1)/2$$

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

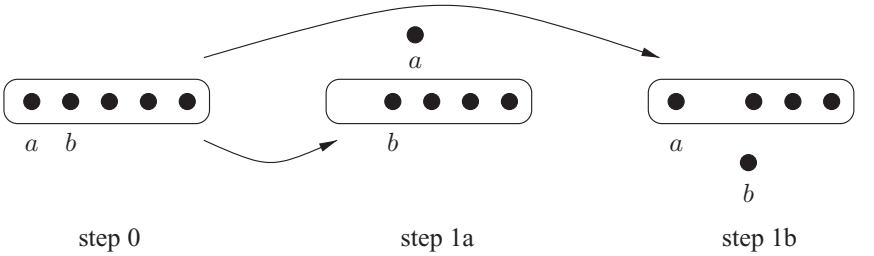


Fig. 1. Alternatives for Stage 1

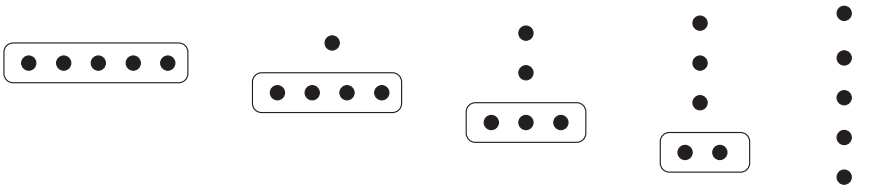


Fig. 2. Stage 1 of refinement (equivalence steps)

steps. That is, the maximum length of a decreasing chain of preorders on P $\sqsubseteq_0, \sqsubseteq_1, \ldots, \sqsubseteq_M$ such that $\sqsubseteq_{k+1}$ is a strict refinement of $\sqsubseteq_k$ is

$$\frac{(N+2)(N-1)}{2}.$$

We will now describe a refinement procedure and compute the number of steps it has, and then prove that a maximal refinement sequence has the same number of elements (steps).

We should obviously start with $\sqsubseteq_0 = P \times P$ (otherwise we could extend our sequence by putting $P \times P$ at the beginning). In the next step we have to remove at least one pair $(a, b) \in \sqsubseteq_0$, and, in order to maintain transitivity, we also need to remove $N-2$ other pairs. They can be either of the form $\{(a, x) \mid x \in P\}$ or $\{(y, b) \mid y \in P\}$ (we can either put a over or b under all the other elements). Both alternatives are shown in Figure 1, where the groups of dots in rounded boxes represent equivalence classes.

Suppose, without loss of generality, that we put a over all elements and that we repeat this procedure of taking one element from the group and putting it *directly above* it until we are left with an N-element chain (Figure 2). Observe that we finish after $N-1$ steps, and in step $i \in \{1, \ldots, N-1\}$, we remove $N-i$ pairs from the previous preorder. Hence, the total number of removed pairs at this stage is

$$1 + \cdots + (N-1) = \frac{N(N-1)}{2}.$$

After obtaining the chain of N elements $\sqsubseteq_{N-1}$, we can proceed by always removing one pair per step. Namely, we take any element x that still has some y above it. Let us choose this y so that it is directly above x and remove (x, y). Observe that we do not have to remove any other pairs since y is directly above x and there are no elements equivalent to x or y. We can do this in every step until we are left with N pairs of the form (x, x).

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Let S_1, S_2 be the number of steps in stages 1 and 2, respectively, and the numbers of pairs removed at each stage be RP_1 and RP_2, respectively. We can derive the total number of steps using the following relationships:

$$\begin{aligned} S_1 &= N - 1 \\ S_2 &= RP_2 \\ RP_1 &= \frac{N(N-1)}{2} \\ N^2 - RP_1 - RP_2 &= N. \end{aligned}$$

So

$$RP_2 = N^2 - N - \frac{N(N-1)}{2} = \frac{N(N-1)}{2},$$

and the total number of steps is

$$S_1 + S_2 = N - 1 + \frac{N(N-1)}{2} = \frac{(N+2)(N-1)}{2}.$$

We still need to prove that we cannot obtain a longer sequence of refinement steps. Consider an *arbitrary* chain of preorders $\sqsubseteq_0, \sqsubseteq_1, \ldots, \sqsubseteq_M$ on an N-element set S such that for all i, we have $\sqsubseteq_i \supset \sqsubseteq_{i+1}$, $\sqsubseteq_0 = S \times S$ and $\sqsubseteq_M = \Delta$, which is the longest chain with these properties.

Elements of such a chain (or refinement steps) can be divided into two categories, as in the example above. We call step i an *equivalence step* if $\sqsubseteq_i$ has more equivalence classes than $\sqsubseteq_{i-1}$, otherwise it is a *simple step*. Both classes of steps generalise the two stages from the example refinement chain above. We want to show that in our optimal chain, the values computed above (S_1, RP_1) and (S_2, RP_2) describe the total number of steps and removed pairs for all equivalence and simple steps, respectively.

Let S_E and RP_E be the total number of steps and removed pairs for equivalence steps, and S_S and RP_S be these numbers for simple steps. We have:

— $S_E = N - 1$
It is obvious that $S_E \leqslant N - 1$ because after at most $N - 1$ equivalence steps, we are left with N equivalence classes. Also, $S_E \geqslant N - 1$, because otherwise there would be a step that would increase the number of equivalence classes with at least 2. In this case, we could always split this single step in two steps and in this way obtain a longer refinement chain.

— $RP_E = \frac{N(N-1)}{2}$
If we consider each element $s \in S$ separately and sum all edges containing s that are removed in equivalence steps, we get $N - 1$. If we sum it for all N nodes, we get a number twice as big as the number of edges removed in all equivalence steps.

— $S_S = RP_S$
Observe that it is optimal to perform simple steps only when there are at least two equivalence classes containing one element each such that one of these equivalence classes is directly above the other. In this case we can perform a simple step that removes only one edge. Otherwise we can perform any available equivalence step.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

In this strategy there is only one edge removed per simple step, and this yields the maximum number of simple steps possible.

In this way, the total number of simple steps is the same as the steps in stage 2 in the example above, and so is the total number of steps in the refinement chain:

$$(N+2)(N-1)/2$$

So we have the following lemma.

Lemma 5. Let S be an N-element set. The maximum length of a decreasing chain of preorders on S $\sqsubseteq_1, \sqsubseteq_2, \ldots$ such that $\sqsubseteq_{k+1}$ is a strict refinement of $\sqsubseteq_k$ is

$$\frac{(N+2)(N-1)}{2}.$$

Corollary 4.

$$D(\sqsubseteq^S, N) \leqslant \frac{(N+2)(N-1)}{2}.$$

We will now show that the upper bound established in Corollary 4 is sharp, that is, for each N, we can find a regular LTS with N states for which the simulation preorder stabilises after only

$$(N+2)(N-1)/2$$

steps. The following algorithm constructs such an LTS for a given N. It requires that there are at least

$$(N+2)(N-1)/2$$

different actions.

1. read N
2. $P := \{p_1, \ldots p_N\}$;
   $\rightarrow_0 := \varnothing$; // $\rightarrow_i$: *transition relation at step i*
   $\preceq_0 := P \times P$; // $\preceq_i$ *maintains preorder at step i*
3. $k := 1$; // *step counter*
   $\rightarrow_1 := \{p_1 \xrightarrow{a_1} p_1\}$
4. $REMOVED_1 := \{(p_1, p_2), (p_1, p_3), \ldots, (p_1, p_N)\}$; $\preceq_1 := \preceq_0 \setminus REMOVED_1$;
5. while $(\preceq_k \neq \Delta)$ // *while current preorder is different from the identity*
   5.1 $k = k + 1$;
   5.2 $(p_{prev\_under}, p_{prev\_over}) :=$ any pair from $REMOVED_{k-1}$
   5.3 $REMOVED_k := \varnothing$; $\rightarrow_k := \rightarrow_{k-1}$; $\preceq_k := \preceq_{k-1}$;
   5.4 for all states $p_i \in P$ add transitions $p_i \xrightarrow{a_k} p_{prev\_over}$ to $\rightarrow_k$, with a_k a fresh action
   5.5 pick any state p_{still_under} such that $\exists_{p_s \in P}\ (p_{still\_under} \preceq_{k-1} p_s)$
   5.6 pick a state $p_{still\_over} \neq p_{still\_under}$ that is either $\preceq_{k-1}$-equivalent to p_{still_under} or, if there are no states equivalent to p_{still_under}, directly above p_{still_under}
   5.7 for all states $p_j \in P$ such that either $(p_j = p_{still\_under})$ or $(p_{still\_under} \preceq_{k-1} p_j$ and p_j is not $\preceq_{k-1}$-equivalent to $p_{still\_over})$, add transitions $p_j \xrightarrow{a_k} p_{still\_under}$ to $\rightarrow_k$;

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

5.8 for all states $p_{s'} \preceq_{k-1}$-equivalent to p_{still_over}
$REMOVED_k := REMOVED_k \cup \{(p_{still\_under}, p_{s'})\}$;
5.9 $\preceq_k := \preceq_k \setminus REMOVED_k$

The algorithm begins with an LTS consisting of N states and no transitions, on which the simulation preorder is equal to $P \times P$. In the first step, only one transition is added, namely $p_1 \xrightarrow{a_1} p_1$. Notice that after the algorithm finishes, all states except for p_1 have the same set of initial actions, whereas p_1 can in addition perform an action a_1. Hence, after the first step, we have an LTS in which p_1 is above all other states in the simulation preorder. The preorder itself is stored in the $\preceq_i$ variable. We will later prove that this variable stores the correct simulation preorder up to i steps for $(P, \rightarrow_n)$, where $n \geqslant i$.

In step $k+1$, for $k \geqslant 1$, the algorithm picks the pair that has been removed from the simulation preorder $(p_{prev\_under}, p_{prev\_over})$ in the previous step. The idea is that if we know that only after $k-1$ steps it is possible to state that p_{prev_under} is not under p_{prev_over} in the simulation preorder, then for any pair $(p_{still\_under}, p_{still\_over})$ from the $\sqsubseteq^S_{k-1}$ preorder, we can add fresh transitions that make p_{still_under} distinguishable from p_{still_over} after precisely k steps. The new transitions must be consistent with the simulation preorder up to $k-1$ steps, and the only inconsistency after k steps should be that p_{still_under} can perform a transition that p_{still_over} (and states equivalent to p_{still_over}) cannot mimic. This transition is

$$p_{still\_under} \xrightarrow{a_k} p_{prev\_under},$$

which in step $k-1$ could be mimicked by

$$p_{still\_over} \xrightarrow{a_k} p_{prev\_over},$$

since

$$p_{prev\_over} \sqsubseteq_{k-2} p_{prev\_under}.$$

We assume a regular LTS $(P, \rightarrow)$ and use the following notion of the *distinguishing depth*:

$$dd(p,q) \stackrel{def}{=} \min\{n \in \mathbb{N} \mid p \not\sqsubseteq^S_n q\}$$

where $\min(\varnothing) = \infty$. We first prove two properties of dd.

Lemma 6. For $p, q \in P$ with $dd(p,q) = k > 1$, there exist $p', q' \in P$ with $dd(p', q') = k-1$ and $a \in Act$ such that $p \xrightarrow{a} p'$ and $q \xrightarrow{a} q'$, and, moreover, there is no $r \in P$ with $p' \sqsubseteq^S_{k-1} r$ such that $q \xrightarrow{a} r$.

Proof. Take any p, q with $dd(p,q) = k$. There is a transition, say $p \xrightarrow{a} p'$, that cannot be simulated by q up to k steps, that is, there is no q' with $q \xrightarrow{a} q'$ and $p' \sqsubseteq^S_{k-1} q'$. Since $p \sqsubseteq^S_{k-1} q$, we know that there is a q' such that $p' \sqsubseteq^S_{k-2} q'$ and $q \xrightarrow{a} q'$. Since $p' \not\sqsubseteq^S_{k-1} q'$, we have $dd(p', q') = k-1$. □

Lemma 7.

$$dd(p,q) = 1 \;\Leftrightarrow\; I(p) \setminus I(q) \neq \varnothing.$$

The following lemma states the crucial properties of the algorithm.

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Lemma 8. The LTS constructed by the algorithm has the following properties:

(1) $\forall (p,q) \in REMOVED_k \, (dd(p,q) = k)$
(2) $\forall (p,q) \in \preceq_k \ (dd(p,q) > k)$.

Proof. We proceed by using induction on k. We have to consider two base cases. For $k = 0$ the $REMOVED_k$ set is empty, hence the first property holds trivially. Moreover, since for any two states p, q we have $dd(p,q) > 0$, we also obtain the second property. For $k = 1$ both properties follow directly from Lemma 7.

We now assume that properties 1 and 2 both hold for all $i \leqslant k$ and prove that they also hold for $k+1$:

(1) We want to prove that $dd(p,q) = k+1$ for all $(p,q) \in REMOVED_{k+1}$. We already know that $dd(p,q) > k$ from the induction hypothesis. Observe that from the construction of the algorithm (Steps 5.2, 5.4 ,5.7), there exist transitions $p \overset{a_{k+1}}{\rightarrow} p'$ and $q \overset{a_{k+1}}{\rightarrow} q'$ with $(p',q') \in REMOVED_k$, so $dd(p',q') = k$ from the induction hypothesis. Since there is only one outgoing transition from q labelled with a_{k+1}, we know that q cannot simulate the transition $p \overset{a_{k+1}}{\rightarrow} p'$ up to more than k steps, hence $p \not\sqsubseteq^S_{k+1} q$ and $dd(p,q) = k+1$.
(2) We prove that for all pairs $(p,q) \in \preceq_{k+1}$, we have $d(p,q) > k+1$. We know from the induction hypothesis that $dd(p,q) > k$, so we need to prove that $dd(p,q) \neq k+1$.
In order to show a contradiction, we suppose that $dd(p,q) = k+1$. Then according to Lemma 6, there would exist (p',q') with $dd(p',q') = k$ and some $i \in \mathbb{N}$ such that $q \overset{a_i}{\rightarrow} q'$, $p \overset{a_i}{\rightarrow} p'$ and $q \overset{a_i}{\not\rightarrow} p'$. Observe that transitions labelled with a_i are added only at step i of the algorithm, and, moreover, if s and s' are two different states with incoming a_i-transitions, then either $(s,s') \in REMOVED_{i-1}$ or $(s',s) \in REMOVED_{i-1}$. If $(q',p') \in REMOVED_{i-1}$, we would also have $q \overset{a_i}{\rightarrow} p'$, because Step 5.4 adds a_i-transitions from all states in P to the second element of the pair. But we have just proved that such a transition cannot exist, so the remaining possibility is that $(p',q') \in REMOVED_{i-1}$.
Let us focus on the value of i. If $i-1 > k$, then $(p',q') \in \preceq_k$, and, from the induction hypothesis, we get $dd(p',q') < k$, which gives a contradiction. If $i-1 \leqslant k$, we can use the induction hypothesis and deduce from $(p',q') \in REMOVED_{i-1}$ that $dd(p',q') = i-1$. We also know that $dd(p',q') = k$, so $i = k+1$. So we have transitions $q \overset{a_{k+1}}{\rightarrow} q'$ and $p \overset{a_{k+1}}{\rightarrow} p'$ with $(p',q') \in REMOVED_k$. Such transitions could only be added if $(p,q) \notin \preceq_{k+1}$, which also gives a contradiction. □

Lemma 9. Let $(P, \rightarrow)$ be the LTS returned by the algorithm. Then for each $k \in \mathbb{N}$, we have $\preceq_k = \sqsubseteq^S_k$, where $\sqsubseteq^S_k$ is the simulation preorder on $(P, \rightarrow)$.

Proof. Take any $k \in \mathbb{N}$. The fact that $\preceq_k \subseteq \sqsubseteq^S_k$ follows directly from the second point in Lemma 8. For the other direction, suppose $(p,q) \notin \preceq_k$. Then (p,q) must have been removed at step $i \leqslant k$, so $(p,q) \in REMOVED_i$ for $i \leqslant k$ and from Lemma 8 (1) it follows that $dd(p,q) \leqslant k$, and hence $(p,q) \notin \sqsubseteq^S_k$. □

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Lemma 10. Upon termination of the algorithm,

$$k = \frac{(N+2)(N-1)}{2}.$$

Proof. Note that the algorithm generates the maximal refinement sequence $\preceq_0, \preceq_1, \ldots, \preceq_N$ starting with $\preceq_0 = P \times P$ and ending with $\preceq_N = \Delta$. Indeed, it has all the properties of the previously discussed optimal refinement procedure: in equivalence steps, it creates exactly two new equivalence classes directly above one another from an existing single one; and in simple steps, it removes exactly one pair from the preorder. □

Theorem 4.

$$D(\sqsubseteq^S, N) = \frac{(N+2)(N-1)}{2}.$$

Proof. We already know that that

$$D(\sqsubseteq^S, N) \leqslant \frac{(N+2)(N-1)}{2}$$

(Corollary 4). For the lower bound, consider the LTS returned by the algorithm. By Lemma 10 and the termination condition in the while loop,

$$\preceq_{\frac{(N+2)(N-1)}{2}-1} \neq \Delta,$$

so, by Lemma 9,

$$\sqsubseteq^S_{\frac{(N+2)(N-1)}{2}-1} \neq \Delta$$

whereas

$$\sqsubseteq^S_{\frac{(N+2)(N-1)}{2}} = \Delta,$$

so there exist states p, q for which it is possible to decide that $p \not\sqsubseteq^S q$ only after

$$\frac{(N+2)(N-1)}{2}$$

steps. □

To derive a corresponding result for simulation equivalence, consider the modified version of an algorithm that keeps two elements in one equivalence class as long as possible. After

$$(N+2)(N-1)/2 - 2$$

steps, we obtain $N-1$ independent equivalence classes, one of which has 2 elements. Therefore it takes a maximum of

$$(N+2)(N-1)/2 - 1$$

steps to remove all non-trivial equivalence classes.

Theorem 5.

$$D(\sim_S, N) = \frac{(N+2)(N-1)}{2} - 1.$$

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

The ready simulation case is very similar. The only difference is that states with different initial actions are incomparable. Therefore in any regular LTS in which not all states are equivalent modulo ready simulation, there are always two incomparable equivalence classes modulo $\sqsubseteq_1^{RS}$.

Theorem 6.

$$D(\sqsubseteq^{RS}, N) = \frac{(N+1)(N-2)}{2} + 1.$$

Proof.

(1) We first prove that

$$D(\sqsubseteq^{RS}, N) \leqslant \frac{(N+1)(N-2)}{2} + 1.$$

We again consider the longest refinement procedure problem, but with the additional restriction that in the first step two incomparable equivalence classes should be created. Notice first that after the first step the existing optimal refinement sequences are applied for each equivalence class separately, so we obtain

$$\frac{(n_1+2)(n_1-1)}{2} + \frac{(n_2+2)(n_2-1)}{2}. \qquad (*)$$

remaining steps, where n_1, n_2 denote the number of elements in each equivalence class. It remains to prove that the greatest value is obtained when one class is a singleton and the other has $N-1$ elements, so we will prove that choosing $n_1 = 1$ maximises the value of (*). Take any partition of elements into two equivalence classes and assume that $n_1 = x_1 + 1$ for $x_1 \geqslant 0$ (we have to assume $n_1 \geqslant 1$) and $n_2 = x_2$ for $x_2 \geqslant 1$. We need to prove

$$\frac{((x_1+x_2)+2)((x_1+x_2)-1)}{2} \geqslant \frac{((x_1+1)+2)((x_1+1)-1)}{2} + \frac{(x_2+2)(x_2-1)}{2}.$$

After elementary algebraic transformations, we get

$$x_1^2 + x_2^2 + 2x_1x_2 + x_1 + x_2 - 2 \geqslant x_1^2 + x_2^2 + 3x_1 + x_2 - 4$$

or, equivalently,

$$x_1x_2 \geqslant x_1 - 1.$$

Since $x_1 \geqslant 0$ and $x_2 \geqslant 1$, the last statement is obviously true. Therefore, the maximum total number of refinement steps is no larger than

$$\frac{((N-1)+2)((N-1)-1)}{2} + 1 = \frac{(N+1)(N-2)}{2} + 1.$$

(2) We now prove

$$D(\sqsubseteq^{RS}, N) \geqslant \frac{(N+1)(N-2)}{2} + 1.$$

We use almost the same algorithm as before to generate the LTS that forces the largest possible number of steps to distinguish a state from another modulo ready simulation. The only difference is that Step 4 should now be replaced by Step 4′:

4′. $REMOVED_1 := \{(p_1, p_j), (p_j, p_1) \mid j \neq 1\}$; $\preceq_1 := \preceq_0 \setminus REMOVED_1$;

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

We can proceed in exactly the same way as for simulation. Lemmas 7, 8 and 9 also hold for the modified algorithm and ready simulation preorder. As for the number of algorithm steps, after step 1, there is one independent state p_1 and an $(N-1)$-element equivalence class, for which it takes

$$\frac{(N+1)(N-2)}{2}$$

steps to finish. Hence the ready simulation algorithm terminates with

$$k = \frac{(N+1)(N-2)}{2} + 1.$$

□

We can obtain the optimal AIP^c value for ready simulation equivalence analogously.

Theorem 7.

$$D(\sim_{RS}, N) = \frac{(N+1)(N-2)}{2}.$$

4.2. *Decorated trace semantics*

Computing preorders on regular LTSs becomes more difficult for decorated trace semantics, which are based on traces of actions that may be augmented with state properties such as allowed/refused actions. Note that deciding trace containment is PSPACE-complete (Stockmeyer and Meyer 1973).

Below, we present a simple property that is common to all decorated trace semantics and guarantees that checking behaviour up to an exponential depth is sufficient to decide whether two states are related by a preorder.

Lemma 11. Assume an LTS $(\mathbb{P}, \rightarrow)$ closed under alternative composition (that is, $S \subseteq \mathbb{P} \Rightarrow \sum_{s\in S} s \in \mathbb{P}$). For any process equivalence $\sim$ satisfying

$$p \sim_{K+1} q \;\Leftrightarrow\; (p \sim_1 q) \wedge \left(\forall_{a\in A} \left(\sum_{p':p\xrightarrow{a}p'} p' \sim_K \sum_{q':q\xrightarrow{a}q'} q'\right)\right). \qquad (\dagger)$$

we have

$$\sim_K = \sim_{K+1} \;\Rightarrow\; \forall i \in \mathbb{N}\,(\sim_K = \sim_{K+i}).$$

Proof. We suppose $\sim_K = \sim_{K+1}$ and take any $p, q \in \mathbb{P}$ such that $p \sim_{K+1} q$. We will prove that $p \sim_{K+2} q$. We have

$$\begin{aligned} p \sim_{K+1} q &\Leftrightarrow (p \sim_1 q) \wedge \forall_{a\in A} \left(\sum_{p':p\xrightarrow{a}p'} p' \sim_K \sum_{q':q\xrightarrow{a}q'} q'\right) && (\dagger)\\ &\Leftrightarrow (p \sim_1 q) \wedge \forall_{a\in A} \left(\sum_{p':p\xrightarrow{a}p'} p' \sim_{K+1} \sum_{q':q\xrightarrow{a}q'} q'\right) && (\sim_K = \sim_{K+1})\\ &\Leftrightarrow p \sim_{K+2} q. && (\dagger) \end{aligned}$$

□

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

Theorem 8. If we assume a regular LTS $(\mathbb{P}, \rightarrow)$ with $|\mathbb{P}| = N$, then for any process equivalence $\sim$ being sound modulo idempotency ($x + x \sim x$), compositional with respect to projection and satisfying

$$p \sim_{K+1} q \;\Leftrightarrow\; (p \sim_1 q) \wedge \left(\forall_{a \in A} \left(\sum_{p' : p \xrightarrow{a} p'} p' \sim_K \sum_{q' : q \xrightarrow{a} q'} q' \right) \right),$$

we have

$$\pi_{\binom{2^N}{2}}(p) \sim \pi_{\binom{2^N}{2}}(q) \;\Rightarrow\; p \sim q,$$

and thus

$$D(\sim, N) \leqslant \binom{2^N}{2}.$$

Proof. First, we extend $\mathbb{P}$ to $\mathbb{P}^+$, which contains all alternative compositions of processes from $\mathbb{P}$. In this way we obtain an LTS with 2^N states. From Lemma 11, we know that if there are processes that can be distinguished after K steps, so in every step i up to K, the equivalence $\sim_i$ is a strict refinement of $\sim_{i+1}$. Since we can have at most $\binom{2^N}{2}$ refinement steps, the upper bound for $D(\sim, N)$. □

It is not difficult to see that this result holds for all decorated trace semantics (trace, completed trace, failures, readiness, failure and ready traces). A sharp bound for AIP^c for these semantics remains an open question.

5. Conclusions and future work

We have presented sufficient conditions for a modal language $\mathcal{O} \subseteq HML$ so that the equivalence $\sim_{\mathcal{O}}$ coincides with its:

(1) finitary counterpart; and
(2) equivalence induced by finite-depth formulae (in the setting of image-finite processes).

We have established that the Approximation Induction Principle holds for all process equivalences whose modal characterisation consists of finite-depth formulae. Combined with the previous result, we have obtained a fairly general sufficient condition for soundness of AIP on image-finite processes. Most 'strong' equivalences in the literature meet the aforementioned conditions.

We have also studied a constructive version of AIP, and proved upper bounds for simulation and ready simulation semantics (quadratic) and for decorated trace semantics (exponential). Constructive AIP gives an upper bound on the number of iterations required for algorithms that compute preorders on, for example, regular LTSs (Celikkan and Cleaveland 1995).

Traditionally, most properties of process semantics were derived by studying each equivalence or preorder separately. The basic idea underlying most results in this paper is to take an orthogonal approach: consider a property and find the class of equivalences that satisfy it. We also expose relationships between the properties. Some recent research on characteristic formulae (Aceto *et al.* 2009) is somewhat similar in spirit to ours in

https://doi.org/10.1017/S0960129511000387 Published online by Cambridge University Press

the sense that a common underlying structure in characteristic formulae constructions for various semantics is explored to create a general framework. Aceto *et al.* consider the domain of behavioural relations that are defined as fixed points of certain monotonic functions, whereas we focus on equivalences defined with modal sublanguages of HML.

We are currently investigating compositionality requirements for process operators. It would also be interesting to consider all these problems in the setting of weak semantics. An extensive list of equivalences involving the silent step along with corresponding modal characterisations has been given in van Glabbeek (1993). The modal languages are more cumbersome, and other issues such as fairness in the case of AIP need to be taken into account.

References

Aceto, L., Bloom, B. and Vaandrager, F. (1994) Turning SOS rules into equations. *Information and Computation* **111** (1) 1–52.

Aceto, L., Ingolfsdottir, A. and Sack, J. (2009) Characteristic formulae for fixed-point semantics: A general framework. Proc. 16th Workshop on Expressiveness in Concurrency (EXPRESS'09). *Electronic Proceedings in Theoretical Computer Science* **8** 1–15.

Baeten, J. C. M., Bergstra, J. A. and Klop, J. W. (1987) On the consistency of Koomen's fair abstraction rule. *Theoretical Computer Science* **51** (1-2) 129–176.

Barros, A. and Hou, T. (2008) A Constructive Version of AIP Revisited. Electronic Report PRG0802, Programming Research Group, University of Amsterdam.

Baeten, J. C. M. and Weijland, W. P. (1990) *Process Algebra*, Cambridge Tracts in Theoretical Computer Science **18**, Cambridge University Press.

Blackburn, P., de Rijke, M. and Venema, Y. (2001) *Modal Logic*, Cambridge University Press.

Bloom, B., Fokkink, W. J. and van Glabbeek, R. J. (2004) Precongruence formats for decorated trace semantics. *ACM Transactions on Computational Logic* **5** (1) 26–78.

Celikkan, U. and Cleaveland, R. (1995) Generating diagnostic information for behavioral preorders. *Distributed Computing* **9** (2) 61–75.

van Glabbeek, R. J. (1987) Bounded nondeterminism and the approximation induction principle in process algebra. Proc. 4th Symposium on Theoretical Aspects of Computer Science (STACS'87). *Springer-Verlag Lecture Notes in Computer Science* **247** 336–347.

van Glabbeek, R. J. (1993) The linear time/branching time spectrum II. The semantics of sequential systems with silent moves. Proc. 4th Conference on Concurrency Theory (CONCUR'93). *Springer-Verlag Lecture Notes in Computer Science* **715** 66–81.

van Glabbeek, R. J. (2001) The linear time/branching time spectrum I. The semantics of concrete, sequential processes. In: Bergstra, J. A., Ponse, A. and Smolka, S. A. (eds.) *Handbook of Process Algebra* 3–99.

van Glabbeek, R. J. and Ploeger, B. (2008) Correcting a space-efficient simulation algorithm. Proc. 20th Conference on Computer Aided Verification (CAV 2008). *Springer-Verlag Lecture Notes in Computer Science* **5123** 517–529.

Goldblatt, R. (1995) Saturation and the Hennessy–Milner property. In: Ponse, A., de Rijke, M. and Venema, Y. (eds.) Modal Logic and Process Algebra. A Bisimulation Perspective. *CSLI Lecture Notes* **53** 189–216.

Hennessy, M. and Milner, R. (1985) Algebraic laws for non-determinism and concurrency. *Journal of the ACM* **32** (1) 137–161.

Hennessy, M. and Milner, R. (1980) On observing nondeterminism and concurrency. Proc. 7th Colloquium on Automata, Languages and Programming (ICALP'80). *Springer-Verlag Lecture Notes in Computer Science* **85** 299–309.

Hollenberg, M. (1995) Hennessy–Milner classes and process algebra. In: Ponse, A., de Rijke, M. and Venema, Y. (eds.) Modal Logic and Process Algebra. A Bisimulation Perspective. *CSLI Lecture Notes* **53** 189–216.

Larsen, K. (1990) Proof systems for satisfiability in Hennessy–Milner logic with recursion. *Theoretical Computer Science* **72** (2-3) 265–288.

Mauw, S. (1987) A constructive version of the approximation induction principle. *Proc. SION Conference on Computing Science in the Netherlands (CSN'87)* 235–252.

Stirling, C. (2001) *Modal and Temporal Properties of Processes*, Texts in Computer Science, Springer-Verlag.

Stockmeyer, L. J. and Meyer, A. R. (1973) Word problems requiring exponential time. *Proc. 5th Annual ACM Symposium on Theory of Computing (STOC'73)* 1–9.