

Modal Logic and the Approximation Induction Principle

Maciej Gazda & Wan Fokkink

Vrije Universiteit
Department of Computer Science
De Boelelaan 1081a, 1081 HV Amsterdam, Netherlands
m.w.gazda@student.vu.nl, wanf@cs.vu.nl

We prove a compactness theorem in the context of Hennessy–Milner logic. It is used to derive a sufficient condition on modal characterizations 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 Approximation Induction Principle with respect to simulation and decorated trace semantics.

1 Introduction

Hennessy–Milner logic [14] is a modal logic for specifying properties of states in a labelled transition system (LTS). Rob van Glabbeek [11] uses this logic to characterize a wide range of process semantics in terms of observations. That is, a process semantics is captured by means of a sublogic of Hennessy–Milner logic; 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 characterizes bisimulation equivalence.

For several process semantics, mainly in the realm of simulation, van Glabbeek introduces three different modal characterizations (see [11, Fig. 9]), which differ in their treatment of conjunction. Apart from the richest characterizations, 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 considered as well (with a superscript ω). The corresponding equivalences all differ in general LTSs and collapse in the setting of image-finite LTSs. An LTS is image-finite if for each state and each action a , there are finitely many outgoing a -transitions. Van Glabbeek sketches separate proofs that the modal characterizations capture the same process semantics under consideration. These proofs are always almost identical.

Here we show that given a modal characterization of a process semantics for general LTSs, restricting to finite sub-conjunctions produces a modal characterization of the same semantics for image-finite LTSs. The only requirement is that the formulas that are thus obtained were already present in the original modal characterization. All semantics in the linear time–branching time spectrum [11] have a modal characterization that satisfies this requirement, except for completed trace semantics (in case of an infinite action set).

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

Van Glabbeek uses a version of Hennessy–Milner logic that contains negation (so that disjunction, falsum, and $[a]\phi$ need not be present). However, in that logic the aforementioned result is not so easy to

obtain. Therefore we first prove the result in a negation-free version of Hennessy–Milner logic. Next we show that the result carries over to Hennessy–Milner logic with negation.

Next we study the Approximation Induction Principle (AIP) from process algebra [3], 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 [9]. Moreover, it is folklore that this soundness result extends to the other equivalences in the linear time–branching time spectrum [1]. We obtain a sufficient condition on the modal characterization of a process equivalence, to guarantee that AIP is sound with respect to this equivalence. The result is then linked to the compactness theorem from the first part. The sufficient condition says that the modal characterization 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 w.r.t. the projection operators used in the definition of AIP, then it can be characterized by a set of finite-depth formulas.

Finally, we consider a so-called constructive version of AIP [18], which states that two processes are equal if they are equal up to *some* finite depth. Constructive AIP has so far 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; moreover, we show that these bounds are sharp. We also provide an exponential bound for decorated trace semantics.

2 Modal Characterizations 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 $s \xrightarrow{a} s'$.

Hennessy–Milner logic [14] is a modal logic for specifying properties of states in an LTS. There exist several versions of Hennessy–Milner logic. The most general language, as presented in [11], is denoted with *HML*. Its syntax can be defined with the following BNF grammar:

$$\varphi ::= \top \mid \bigwedge_{i \in I} \varphi_i \mid \langle a \rangle \varphi \mid \neg \varphi$$

The meaning of the formulas is defined inductively as follows:

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

There exists a different syntax (see [17], [19]) of Hennessy–Milner logic without negation symbol, denoted with *HML*⁺. As we will see later on, its formulas have nice properties which make it easier to perform certain proofs.

$$\phi ::= \top \mid \text{F} \mid \bigwedge_{i \in I} \phi_i \mid \bigvee_{i \in I} \phi_i \mid \langle a \rangle \phi \mid [a] \phi$$

The meaning of the new formulas is defined below:

$$s \not\models \text{F} \quad s \models \bigvee_{i \in I} \phi_i \Leftrightarrow \exists i \in I (s \models \phi_i) \quad s \models [a] \phi \Leftrightarrow \forall s' \in S (s \xrightarrow{a} s' \Rightarrow s' \models \phi)$$

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 depth of a formula $d : HML \longrightarrow \mathbb{N} \cup \{\infty\}$ inductively as:

$$d(\top) = 0 \quad d(\bigwedge_{i \in I} \phi_i) = \sup\{d(\phi_i) \mid i \in I\} \quad d(\langle a \rangle \phi) = 1 + d(\phi) \quad d(-\phi) = d(\phi)$$

(where $1 + \infty = \infty$). HML_{FDP} and HML_{FDP}^+ denote classes of formulas of finite depth: $HML_{FDP}^{(+)} = \{\phi \in HML^{(+)} \mid d(\phi) < \infty\}$.

A *context* $C[\]$ denotes a formula containing one occurrence of \square . The formula $C[\phi]$ is obtained by replacing this occurrence of \square 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 $\phi \Rightarrow \psi$ denotes that for any state s , $s \models \phi \Rightarrow s \models \psi$).

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.

In [17] there is a result (Lem. 2.8) which implies the proposition below, but only for HML_{FIN}^+ formulas. Moreover, in [17] no proof is provided for Lem. 2.8. Therefore we include a proof of Prop. 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 $J \subseteq_{\text{FIN}} I$, $\bigwedge_{i \in I} \phi_i \Rightarrow \bigwedge_{i \in J} \phi_i$, and so $C[\bigwedge_{i \in I} \phi_i] \Rightarrow C[\bigwedge_{i \in J} \phi_i]$.

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

- $C[\] = \square$.

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 $s \models C'[\bigwedge_{i \in I} \phi_i]$. Hence $s \models C[\bigwedge_{i \in I} \phi_i]$.

- $C[\] = C'[\] \vee \bigvee_{k \in K} \psi_k$.

If $s \models \psi_{k_0}$ for some $k_0 \in K$, then clearly $s \models C[\bigwedge_{i \in I} \phi_i]$. Suppose then $s \models C'[\bigwedge_{i \in J} \phi_i]$ for all $J \subseteq_{\text{FIN}} I$. Then by induction $s \models C'[\bigwedge_{i \in I} \phi_i]$, and 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 $\{s_{J_1}, \dots, s_{J_m}\}$. Suppose, towards a contradiction, that $s_{J_k} \not\models C'[\bigwedge_{i \in I} \phi_i]$ for all $k = 1, \dots, m$. Then by the inductive assumption, for all $k = 1, \dots, m$, $s_{J_k} \not\models C'[\bigwedge_{i \in K_k} \phi_i]$ for some $K_k \subseteq_{\text{FIN}} I$. This implies that, for all $k = 1, \dots, m$, $s_{J_k} \not\models C'[\bigwedge_{\ell=1}^m \bigwedge_{i \in K_\ell} \phi_i]$. This contradicts the fact that $s_{\cup_{\ell=1}^m K_\ell} \in \{s_{J_1}, \dots, s_{J_m}\}$. We conclude that $s_{J_{k_0}} \models C'[\bigwedge_{i \in I} \phi_i]$ for some $k_0 \in \{1, \dots, m\}$. 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 $s \models [a]C'[\bigwedge_{i \in I} \phi_i]$. □

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

There is a counterpart of Prop. 1, for disjunction instead of conjunction. To derive this lemma immediately from Prop. 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 $\bar{\phi} \in HML^+$ is defined inductively as follows:

$$\begin{array}{lll} \bar{\top} = \text{F} & \overline{\bigwedge_{i \in I} \phi_i} = \bigvee_{i \in I} \bar{\phi}_i & \overline{\langle a \rangle \phi} = [a] \bar{\phi} \\ \bar{\text{F}} = \text{T} & \overline{\bigvee_{i \in I} \phi_i} = \bigwedge_{i \in I} \bar{\phi}_i & \overline{[a] \phi} = \langle a \rangle \bar{\phi} \end{array}$$

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

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

Proof: $s \models C[\bigvee_{i \in I} \phi_i] \Leftrightarrow s \not\models \overline{C[\bigvee_{i \in I} \phi_i]} \Leftrightarrow s \not\models \bar{C}[\bigwedge_{i \in I} \bar{\phi}_i] \Leftrightarrow s \not\models \bar{C}[\bigwedge_{i \in J} \bar{\phi}_i]$ for some $J \subseteq_{\text{FIN}} I$ (by Prop. 1) $\Leftrightarrow s \not\models \overline{C[\bigvee_{i \in J} \phi_i]}$ for some $J \subseteq_{\text{FIN}} I \Leftrightarrow s \models C[\bigvee_{i \in J} \phi_i]$ for some $J \subseteq_{\text{FIN}} I$. □

Now we move to Hennessy–Milner logic with negation, HML . Contexts over this syntax are denoted by $D[]$. Each formula ϕ over this logic can be translated to an equivalent formula $P(\phi) \in HML^+$ in a straightforward fashion:

$$\begin{array}{ll} P(\top) & = \top & P(\bigwedge_{i \in I} \phi_i) & = \bigwedge_{i \in I} P(\phi_i) \\ P(\langle a \rangle \phi) & = \langle a \rangle P(\phi) & P(\neg \phi) & = \overline{P(\phi)} \end{array}$$

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

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

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

Lemma 1 $P(D[\phi]) = P(D)[P(\phi)]$ if $D[]$ is a positive context, and $P(D[\phi]) = P(D)[\overline{P(\phi)}]$ if $D[]$ is a negative context.

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

First let $D[]$ be positive, so $D'[]$ is negative. Then $P(\neg D'[\phi]) = \overline{P(D'[\phi])} = \overline{P(D')[\overline{P(\phi)}]}$ (by induction) $= \overline{P(D')[\overline{P(\phi)}]} = P(\neg D')[P(\phi)]$.

Next let $D[]$ be negative, so $D'[]$ is positive. Then $P(\neg D'[\phi]) = \overline{P(D'[\phi])} = \overline{P(D')[P(\phi)]}$ (by induction) $= \overline{P(D')[P(\phi)]} = P(\neg D')[P(\phi)]$. □

Now we can prove a counterpart of Prop. 1 and 2 for *HML*.

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 $J \subseteq_{\text{FIN}} I$.
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 $J \subseteq_{\text{FIN}} I$.

Proof: If $D[\]$ is a positive context, then

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

If $D[\]$ is a negative context, then

$$\begin{aligned} s \models D[\bigwedge_{i \in I} \varphi_i] &\Leftrightarrow s \models P(D[\bigwedge_{i \in I} \varphi_i]) \Leftrightarrow s \models P(D)[\bigvee_{i \in I} \overline{P(\varphi_i)}] \text{ (by Lem. 1)} \Leftrightarrow \\ s \models P(D)[\bigvee_{i \in J} \overline{P(\varphi_i)}] &\text{ for some } J \subseteq_{\text{FIN}} I \text{ (by Prop. 2)} \Leftrightarrow s \models P(D)[\bigwedge_{i \in J} \varphi_i] \text{ for some } J \subseteq_{\text{FIN}} I \text{ (by Lem. 1)} \\ &\Leftrightarrow s \models D[\bigwedge_{i \in J} \varphi_i] \text{ for some } J \subseteq_{\text{FIN}} I. \quad \square \end{aligned}$$

2.3 Modal Characterizations

A process semantics on LTSs can be captured by means of a sublogic of *HML*; see [7, 11] for a wide range of such modal characterizations. Given such a sublogic \mathcal{O} , two states in an LTS are equivalent if and only if they make true exactly the same formulas in \mathcal{O} . We denote this equivalence relation on states by $\sim_{\mathcal{O}}$.

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

We obtain a similar compactness result for modal characterizations of 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, the original and the resulting modal characterization coincide, if the resulting formulas were already present in the original modal characterization.

The modal characterizations in [7] all satisfy this requirement, except for the one of completed trace semantics, in case of an infinite action set. Namely, the modal characterization of completed trace semantics, for general LTSs as well as for image-finite ones, is:

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

where A denotes the set of all actions.

Given a modal characterization \mathcal{O} , we denote the sublogic of formulas in \mathcal{O} that do not contain infinite conjunctions by \mathcal{O}_{FIN} and the sublogic of formulas with finite depth with \mathcal{O}_{FDP} . Clearly $\mathcal{O}_{\text{FIN}} \subseteq \mathcal{O}_{\text{FDP}}$. Using the results from Sect. 2.2, we can now prove the main theorem of this section.

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

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

Proof: We prove the theorem for the subset of finite formulas \mathcal{O}_{FIN} , and make remarks between square brackets about the version with \mathcal{O}_{FDP} whenever it is necessary. Since $\mathcal{O}_{\text{FIN}} \subseteq \mathcal{O}_{\text{FDP}} \subseteq \mathcal{O}$, clearly $\sim_{\mathcal{O}} \subseteq \sim_{\mathcal{O}_{\text{FDP}}} \subseteq \sim_{\mathcal{O}_{\text{FIN}}}$. We need to show that \mathcal{O}_{FIN} [resp. \mathcal{O}_{FDP}] can distinguish all states that \mathcal{O} can.

Given states s, s' and a formula $\varphi \in \mathcal{O}$ with $s \models \varphi$ and $s' \not\models \varphi$. We construct a formula in \mathcal{O}_{FIN} [resp. \mathcal{O}_{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,

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

The base case is trivial, because if $\lambda(\varphi) = 0$, then $\varphi \in \mathcal{O}_{\text{FIN}}$ [resp. $\varphi \in \mathcal{O}_{\text{FDP}}$]. 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 outermost, in the sense that it does not occur within any infinite conjunction [of infinite depth]. We distinguish two cases.

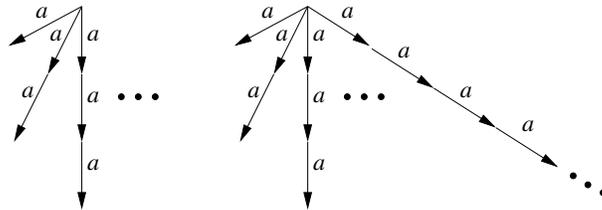
- $D[\]$ is a positive context. By Prop. 3.1, $s' \not\models \varphi$ implies that $s' \not\models D[\bigwedge_{i \in J_0} \varphi_i]$ for some $J_0 \subseteq_{\text{FIN}} I$, while $s \models \varphi$ implies that $s \models D[\bigwedge_{i \in J_0} \varphi_i]$.
- $D[\]$ is a negative context. By Prop. 3.2, $s \models \varphi$ implies that $s \models D[\bigwedge_{i \in J_0} \varphi_i]$ for some $J_0 \subseteq_{\text{FIN}} I$, while $s' \not\models \varphi$ implies that $s' \not\models D[\bigwedge_{i \in J_0} \varphi_i]$.

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

Clearly, 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 $\lambda(\psi) < \lambda(\varphi)$, by ordinal induction, we can construct a formula in \mathcal{O}_{FIN} [resp. \mathcal{O}_{FDP}] that distinguishes s and s' . \square

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

The following example, taken from [9], shows that Thm. 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 $\mathcal{O} = \{\langle a \rangle(\bigwedge_{n \in \mathbb{N}} \langle a \rangle^n \top) \mid N \subseteq \mathbb{N}\}$. Then $\mathcal{O}_{\text{FIN}} = \{\langle a \rangle(\bigwedge_{n \in N} \langle a \rangle^n \top) \mid N \subseteq_{\text{FIN}} \mathbb{N}\}$. Clearly, \mathcal{O} distinguishes the top states of the two LTSs above, by means of any formula $\langle a \rangle(\bigwedge_{n \in N} \langle a \rangle^n \top)$ 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}_{FIN} does not distinguish these states; all formulas in \mathcal{O}_{FIN} hold for both states.

Goldblatt [13] and Hollenberg [16] (see also [6]) investigated models that are more general than image-finite LTSs, but that 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 , then there exists an a -successor of s in which \mathcal{O} is satisfied. It is not difficult to prove, with ordinal induction on the structure of formulas, that Thm. 1 holds for M-saturated models as well.

3 Approximation Induction Principle

For each natural number n we define a *projection operator* π_n which mimicks 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.

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

3.1 Sufficient Criterion for Soundness of AIP

In [1] it is stated that AIP is sound for all 11 ”strong” equivalences from [11], but no argument is provided. Soundness of AIP has been proved several times for bisimulation equivalence (e.g. [9]) 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 that one of the processes is image-finite). A different proof has been presented in [5]. 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 n -th projection are linked with those bisimulations for $(n+1)$ -th projection in which they are included. 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. Namely, AIP is sound for all process equivalences that can be defined using modal characterizations 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.

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

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

Proof: We proceed with ordinal induction on the complexity of a formula, defined by:

$$|\top| = 1 \quad |\langle a \rangle \varphi| = |\varphi| + 1 \quad \left| \bigwedge_{i \in I} \varphi_i \right| = \sup\{|\varphi_i| \mid i \in I\} + 1 \quad |\neg \varphi| = |\varphi| + 1$$

” \Rightarrow ”: The base is trivial ($\varphi = \top$). Let s be an arbitrary state, φ a formula such that $s \models \varphi$, and suppose that for all s' and for all ψ with $|\psi| < |\varphi|$, $s' \models \psi$ implies that ψ is satisfied by all projections $\pi_n(s')$ for $n \geq d(\psi)$. 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: $\forall n \geq d(\psi) (\pi_n(q) \models \psi)$. Since $\pi_n(s) \xrightarrow{a} \pi_{n-1}(q)$ for $n \geq 1$, we have: $\forall n \geq d(\psi) + 1 (\pi_n(s) \models \langle a \rangle \psi)$, so $\forall n \geq d(\langle a \rangle \psi) (\pi_n(s) \models \langle a \rangle \psi)$

- $\varphi = \bigwedge_{i \in I} \psi_i$

Then $\forall i \in I (s \models \psi_i)$. By induction, this implies: $\forall i \in I \forall n \geq d(\psi_i) (\pi_n(s) \models \psi_i)$. Therefore $\forall n \geq \max_{i \in I} \{d(\psi_i)\} \forall i \in I (\pi_n(s) \models \psi_i)$. By definition $d(\bigwedge_{i \in I} \psi_i) = \max_{i \in I} \{d(\psi_i)\}$, so $\forall n \geq d(\bigwedge_{i \in I} \psi_i) (\pi_n(s) \models \bigwedge_{i \in I} \psi_i)$.

- $\varphi = \neg \psi$

We have to consider all the subcases, depending on ψ :

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

” \Leftarrow ”: The other direction follows immediately from what we have just proven. Take an arbitrary formula $\varphi \in \mathcal{O}$ and a state s such that $\forall n \geq d(\varphi) (\pi_n(s) \models \varphi)$. Suppose towards a contradiction that $s \not\models \varphi$. Then $s \models \neg \varphi$, and it was already proven that this implies $\forall n \geq d(\neg \varphi) (\pi_n(s) \models \neg \varphi)$. This contradicts our assumptions. Therefore s must satisfy φ . \square

Theorem 2 If $\mathcal{O} \subseteq HML_{FDP}$, then 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 that $\forall n \in N(\pi_n(s) \sim_{\mathcal{O}} \pi_n(q))$. We have to prove that $\mathcal{O}(s) = \mathcal{O}(q)$. In fact it suffices to prove that $\mathcal{O}(s) \subseteq \mathcal{O}(q)$, the proof of the other inclusion is symmetric. Take any $\varphi \in \mathcal{O}(s)$. According to the Lemma 2, $\forall n \geq d(\varphi) (\varphi \in \mathcal{O}(\pi_n(s)) = \mathcal{O}(\pi_n(q)))$. Using the same lemma again we obtain $\varphi \in \mathcal{O}(q)$. \square

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

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 $d(\bigwedge_{i \in I} \varphi_i) = \infty$, $D[\bigwedge_{i \in J} \varphi_i] \in \mathcal{O}$ for all $J \subseteq_{\text{FIN}} I$. 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 Thm. 1.2 $\sim_{\mathcal{O}} = \sim_{\mathcal{O}'}$, where $\mathcal{O}' \in HML_{FDP}$. By Thm. 2 AIP is sound for $\sim_{\mathcal{O}}$. \square

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 ($n \geq 1$), bisimulation.

Proof: As pointed in [11], all the above equivalences with the exception of completed trace can be defined with a sublogic of Hennessy–Milner logic consisting of finite formulas. Moreover, all formulas in the modal language corresponding to completed trace equivalence are finite-depth. \square

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") with 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 $E(\langle a \rangle T)$ is satisfied). It is easy to observe 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 which are compositional w.r.t. 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 with F . The functions $cut_n : HML \rightarrow HML_{FDP}$ for $n \in \mathbb{N}$ are defined inductively as follows:

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

We now prove a key property for cut functions.

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

$$(CT) \quad \pi_n(s) \models \varphi \Leftrightarrow s \models cut_n(\varphi)$$

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

- $\varphi = T$:
 $\pi_n(s) \models T$ and $s \models cut_n(T) = T$.
- $\varphi = \langle a \rangle \psi$: Now we distinguish cases where $n = 0$ and $n > 0$.

Clearly $\pi_0(s) \not\models \langle a \rangle \psi$ and $s \not\models cut_0(\langle a \rangle \psi) = F$.

If $n > 0$, then $\pi_n(s) \models \langle a \rangle \psi \Leftrightarrow \exists s' (s \xrightarrow{a} s' \wedge \pi_{n-1}(s') \models \psi)$ (transition rules for π_{n-1}) $\Leftrightarrow \exists s' (s \xrightarrow{a} s' \wedge s' \models cut_{n-1}(\psi))$ (structural induction) $\Leftrightarrow s \models \langle a \rangle cut_{n-1}(\psi) \Leftrightarrow s \models cut_n(\langle a \rangle \psi)$ (definition of cut)

- $\varphi = \bigwedge_{i \in I} \psi_i$:
 $\pi_n(s) \models \bigwedge_{i \in I} \psi_i \Leftrightarrow \forall i \in I (\pi_n(s) \models \psi_i) \Leftrightarrow \forall i \in I (s \models cut_n(\psi_i))$ (structural induction) $\Leftrightarrow s \models cut_n(\bigwedge_{i \in I} \psi_i)$ (definition of cut)
- $\varphi = \neg\psi$:
 $\pi_n(s) \models \neg\psi \Leftrightarrow \pi_n(s) \not\models \psi \Leftrightarrow s \not\models cut_n(\psi)$ (structural induction) $\Leftrightarrow s \models \neg cut_n(\psi) \Leftrightarrow s \models cut_n(\neg\psi)$ (definition of cut)

□

Theorem 3 Suppose $\sim_{\mathcal{O}}$ is a process equivalence induced by some $\mathcal{O} \subseteq HML$ and compositional w.r.t. 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 ": That definability of an equivalence with a sublogic of HML_{FDP} implies soundness of AIP has been already proven in Thm. 2.

" \Rightarrow ": We have to prove that soundness of AIP implies $\exists \mathcal{O}_1 \subseteq HML_{FDP} (s \sim_{\mathcal{O}} q \Leftrightarrow s \sim_{\mathcal{O}_1} q)$. 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:

$$\begin{aligned}
s \sim_{\mathcal{O}} q &\Leftrightarrow \forall n \in \mathbb{N} (\pi_n(s) \sim_{\mathcal{O}} \pi_n(q)) \text{ (soundness of AIP for } \sim_{\mathcal{O}} \text{ \& compositional w.r.t. projection)} \\
&\Leftrightarrow \forall n \in \mathbb{N} \forall \varphi \in \mathcal{O} (\pi_n(s) \models \varphi \Leftrightarrow \pi_n(q) \models \varphi) \\
&\Leftrightarrow \forall n \in \mathbb{N} \forall \varphi \in \mathcal{O} (s \models cut_n(\varphi) \Leftrightarrow q \models cut_n(\varphi)) \text{ (Lem. 3)} \\
&\Leftrightarrow \forall n \in \mathbb{N} \forall \psi \in \mathcal{O}_1 (s \models \psi \Leftrightarrow q \models \psi) \text{ (def. of } \mathcal{O}_1) \\
&\Leftrightarrow \forall n \in \mathbb{N} (\pi_n(s) \sim_{\mathcal{O}_1} \pi_n(q)) \Leftrightarrow s \sim_{\mathcal{O}_1} q \quad \square
\end{aligned}$$

4 Constructive AIP

The Approximation Induction Principle is not very attractive in the computational sense, because 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 by Mauw in [18] and investigated further in [4]. AIP^c makes an additional assumption that LTSs are regular (i.e. have a finite number of states and transitions). Such LTSs can be defined with n linear equations for some $n \in \mathbb{N}$; each equation defines a state and its outgoing transitions. AIP^c specifies a value $K \in \mathbb{N}$ for which equivalence between two states can be decided by comparing their K -th projections. Constructive AIP has so far only been considered for bisimulation equivalence, for which it has been proved that if $\pi_{n-1}(p) \Leftrightarrow \pi_{n-1}(q)$, then $p \Leftrightarrow q$, where n is the number of linear equations defining an LTS that contains p and q [4].

We investigate the possibility of obtaining similar results for other process semantics. In this section our approach is different in two ways: we focus on some particular semantics separately, and we consider AIP^c also 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 this moment on, we only consider regular LTSs. For a preorder \sqsubseteq , by $D(\sqsubseteq, N)$ we denote the smallest number such that for all regular LTSs with N distinct states, $\pi_{D(\sqsubseteq, N)}(p) \sqsubseteq \pi_{D(\sqsubseteq, N)}(q) \Rightarrow p \sqsubseteq q$. We introduce a similar notation $D(\sim, N)$ for equivalences.

Assume a preorder (P, \preceq) and let us denote with \prec the corresponding strong order relation, i.e. $x \prec y$ if $x \preceq y$ and $x \neq y$. We say that b is *directly above* a iff $a \prec b$ 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 $b \preceq a$. By Δ we denote an identity relation.

4.1 Simulation-based Semantics

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

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

We define the [ready] simulation preorder as: $p \sqsubseteq^S q$ [resp. $p \sqsubseteq^{RS} q$] if there exists a [ready] simulation relation R such that pRq .

For $X \in \{S, RS\}$ and $K \geq 0$, let \sqsubseteq_K^X denote the [ready] simulation preorder up to K steps: $p \sqsubseteq_K^X q$ iff $\pi_K(p) \sqsubseteq^X \pi_K(q)$.

Lemma 4 For $X \in \{S, RS\}$, if $\sqsubseteq_{K+1}^X = \sqsubseteq_K^X$ for some $K \in \mathbb{N}$, then $\sqsubseteq_N^X = \sqsubseteq_K^X$ for all $N > K$.

Proof: We present a proof for simulation semantics with additional remarks for ready simulation. Suppose, towards contradiction, that there exists an $N > K + 1$ such that $\sqsubseteq_N^S \neq \sqsubseteq_K^S$. Let N be the lowest number with this property. Since clearly $\sqsubseteq_N^S \subseteq \sqsubseteq_{N-1}^S$, there must exist processes p, q such that $p \sqsubseteq_{N-1}^S q$, but $p \not\sqsubseteq_N^S q$. Therefore there exists a transition $p \xrightarrow{a} p'$ that cannot be mimicked by q , i.e. $\neg \exists q' (q \xrightarrow{a} q' \wedge p' \sqsubseteq_{N-1}^S q')$ [in the ready simulation case, we have to consider the case when $I(p) \neq I(q)$, but this is impossible since $p \sqsubseteq_{N-1}^{RS} q$ and $N - 1 \geq 1$]. Since $p \sqsubseteq_{N-1}^S q$, we know that $\exists q' (q \xrightarrow{a} q' \wedge p' \sqsubseteq_{N-2}^S q')$ and since $\sqsubseteq_{N-1}^S = \sqsubseteq_{N-2}^S$, we have $p' \sqsubseteq_{N-1}^S q'$, a contradiction. \square

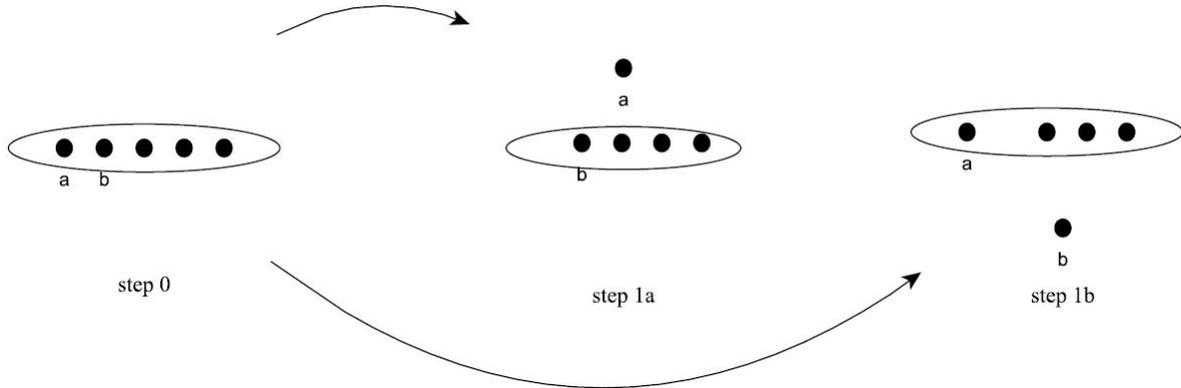
Since $\sqsubseteq^S \subseteq \sqsubseteq_{K+1}^S \subseteq \sqsubseteq_K^S$ for all $K \in \mathbb{N}$, \sqsubseteq_0^S relates all N^2 possible pairs, and \sqsubseteq^S includes the N identity pairs, we obtain the following:

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

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

$\frac{(N+2)(N-1)}{2}$. To explain the ideas underlying AIP^c for the (ready) simulation preorder, we first prove that $D(\sqsubseteq^S, N) \leq \frac{(N+2)(N-1)}{2}$. This is an immediate consequence of a more general preorder property, namely for any preorder (P, \sqsubseteq) with $|P| = N$, we can refine \sqsubseteq in at most $\frac{(N+2)(N-1)}{2}$ steps. In other words, the maximum length of a decreasing chain of preorders on P $\sqsubseteq_0, \sqsubseteq_1, \dots, \sqsubseteq_M$ such that \sqsubseteq_{k+1} is a strict refinement of \sqsubseteq_k is $\frac{(N+2)(N-1)}{2}$. We now describe a refinement procedure, compute its number of steps, 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 below (encircled groups of dots 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 (Fig.1). Observe that we finish after $N - 1$ steps, and in step $i \in \{1, \dots, N - 1\}$ we remove $N - i$ pairs from the previous preorder. Hence the total number of removed pairs at this stage is $1 + \dots + (N - 1) = \frac{N(N-1)}{2}$.

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 which still has some y above it. Let us choose this y so that it is directly

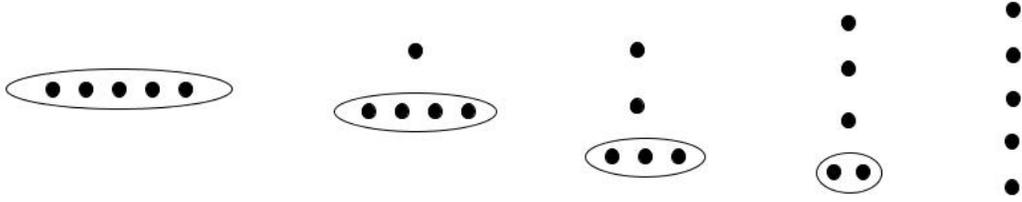


Figure 1: Stage 1 of refinement (equivalence steps)

above x and remove (x, y) . Observe that we do not have to remove any other pairs, because 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) .

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

- $S_1 = N - 1$
- $S_2 = RP_2$
- $RP_1 = \frac{N(N-1)}{2}$
- $N^2 - RP_1 - RP_2 = N \Rightarrow RP_2 = N^2 - N - \frac{N(N-1)}{2} = \frac{N(N-1)}{2}$

Therefore the total number of steps is: $S_1 + S_2 = N - 1 + \frac{N(N-1)}{2} = \frac{(N+2)(N-1)}{2}$

What remains is to prove that we cannot obtain a longer sequence of refinement steps. Consider an *arbitrary* chain of preorders $\sqsubseteq_0, \sqsubseteq_2, \dots, \sqsubseteq_M$ on an N -element set S such that for all i , $\sqsubseteq_i \supset \sqsubseteq_{i+1}$, $\sqsubseteq_0 = S \times S$ 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, like 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 generalize 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 us denote with S_E and RP_E the total number of steps and removed pairs for equivalence steps, in case of simple steps we use S_S and RP_S . We have:

- $S_E = N - 1$

Obviously $S_E \leq N - 1$, because after at most $N - 1$ equivalence steps we are left with N equivalence classes. Also, $S_E \geq N - 1$, 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 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 which removes only one edge. Otherwise we can perform any available equivalence step. In this strategy there is only one edge removed per simple step, and this yields the maximum number of simple steps possible.

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: $\frac{(N+2)(N-1)}{2}$. We thus have the following:

Lemma 5 Let S be an N -element set. The maximum length of a decreasing chain of preorders on $\sqsubseteq_1, \sqsubseteq_2, \dots$ such that \sqsubseteq_{k+1} is a strict refinement of \sqsubseteq_k is $\frac{(N+2)(N-1)}{2}$.

Corollary 4 $D(\sqsubseteq^S, N) \leq \frac{(N+2)(N-1)}{2}$.

We now show that the upper bound established in Corollary 4 is sharp, i.e. for each N we can find a regular LTS with N states for which the simulation preorder stabilizes only after $\frac{(N+2)(N-1)}{2}$ steps. The following algorithm constructs such an LTS for a given N . It requires that there are at least $\frac{(N+2)(N-1)}{2}$ different actions.

1. read N
2. $P := \{p_1, \dots, p_N\}$;
 $\rightarrow_0 := \emptyset$; // \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. $REMOVED_1 := \{(p_1, p_2), (p_1, p_3), \dots, (p_1, p_N)\}$; $\preceq_1 := \preceq_0 \setminus REMOVED_1$;
5. while ($\preceq_k \neq \Delta$) // while current preorder is different than identity
 - 5.1 $k = k + 1$;
 - 5.2 $(p_{prev_under}, p_{prev_over}) :=$ any pair from $REMOVED_{k-1}$
 - 5.3 $REMOVED_k := \emptyset$; $\rightarrow_k := \rightarrow_{k-1}$; $\preceq_k := \preceq_{k-1}$;
 - 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 $p_{still_over} \neq p_{still_under}$ which 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 ;
 - 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 $n \geq i$.

In step $k + 1$, for $k \geq 1$, 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_{k-1}^S preorder, we can add fresh transitions which 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 which 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 $p_{prev_over} \sqsubseteq_{k-2} p_{prev_under}$.

Assume a regular LTS (P, \rightarrow) . We use the following notion of *distinguishing depth*:

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

(where $\min(\emptyset) = \infty$). 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 $p' \sqsubseteq_{k-1}^S r$ such that $q \xrightarrow{a} r$.

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

Lemma 7 $dd(p, q) = 1 \Leftrightarrow I(p) \setminus I(q) \neq \emptyset$

The following lemma states the crucial properties of the algorithm.

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 with 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 Lem. 7.

Assume now that properties 1 and 2 both hold for all $i \leq k$. We prove that they also hold for $k + 1$.

1. We want to prove that for all $(p, q) \in REMOVED_{k+1}$, $dd(p, q) = k + 1$. We already know that $dd(p, q) > k$ from the induction hypothesis. Observe that from the construction of the algorithm (points 5.2, 5.4, 5.7) there exist transitions $p \xrightarrow{a_{k+1}} p'$ and $q \xrightarrow{a_{k+1}} q'$ with $(p', q') \in REMOVED_k$, and hence $dd(p', q') = k$ from the induction hypothesis. Since there is only one outgoing transition from q labelled with a_{k+1} , q cannot simulate the transition $p \xrightarrow{a_{k+1}} p'$ up to more than k steps, hence $p \not\sqsubseteq_{k+1}^S q$ and $dd(p, q) = k + 1$.
2. We prove that for all pairs $(p, q) \in \preceq_{k+1}$, $dd(p, q) > k + 1$. We know from the induction hypothesis that $dd(p, q) > k$, so we need to prove that $dd(p, q) \neq k + 1$.

Suppose towards a contradiction that $dd(p, q) = k + 1$. Then according to Lem. 6 there would exist (p', q') with $dd(p', q') = k$ and some $i \in \mathbb{N}$ such that $q \xrightarrow{a_i} q'$, $p \xrightarrow{a_i} p'$ and $q \not\rightarrow p'$. 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 $(q', p') \in \text{REMOVED}_{i-1}$, then we would also have $q \xrightarrow{a_i} p'$, because step 5.4 adds a_i -transitions from all states in P to the second element of the pair. But we just proved that such transition cannot exist, so the remaining possibility is that $(p', q') \in \text{REMOVED}_{i-1}$.

Let us focus on the value of i . If $i - 1 > k$, then $(p', q') \in \preceq_k$, and from the induction hypothesis it follows that $dd(p', q') < k$, a contradiction. If $i - 1 \leq k$, then we can use the induction hypothesis and deduce from $(p', q') \in \text{REMOVED}_{i-1}$ that $dd(p', q') = i - 1$. We also know that $dd(p', q') = k$, so $i = k + 1$. We thus have transitions $q \xrightarrow{a_{k+1}} q'$ and $p \xrightarrow{a_{k+1}} p'$ with $(p', q') \in \text{REMOVED}_k$. Such transitions could only be added if $(p, q) \notin \preceq_{k+1}$, a contradiction. \square

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

Proof: Take any $k \in \mathbb{N}$. That $\preceq_k \subseteq \sqsubseteq_k^S$ follows directly from the second point in Lem. 8. For the other direction, suppose that $(p, q) \notin \preceq_k$. Then (p, q) must have been removed at step $i \leq k$, so $(p, q) \in \text{REMOVED}_i$ for $i \leq k$ and from the first point of Lem. 8 it follows that $dd(p, q) \leq k$ and hence $(p, q) \notin \sqsubseteq_k^S$. \square

Lemma 10 Upon termination of the algorithm, $k = \frac{(N+2)(N-1)}{2}$.

Proof: Observe that the algorithm generates the maximal refinement sequence $\preceq_0, \preceq_1, \dots, \preceq_N$ 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. \square

Theorem 4 $D(\sqsubseteq^S, N) = \frac{(N+2)(N-1)}{2}$.

Proof: We already know that $D(\sqsubseteq^S, N) \leq \frac{(N+2)(N-1)}{2}$ (Corollary 4). For the lower bound, consider the LTS returned by the algorithm. By Lem. 10 and the termination condition in the while loop, $\preceq_{\frac{(N+2)(N-1)}{2} - 1} \neq \Delta$, and therefore by Lem. 9, $\sqsubseteq_{\frac{(N+2)(N-1)}{2} - 1}^S \neq \Delta$ whereas $\sqsubseteq_{\frac{(N+2)(N-1)}{2}}^S = \Delta$, 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. \square

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 $\frac{(N+2)(N-1)}{2} - 2$ steps we obtain $N - 1$ independent equivalence classes, one of which has 2 elements. Therefore it takes a maximum of $(\frac{(N+2)(N-1)}{2} - 1)$ steps to remove all non-trivial equivalence classes.

Theorem 5 $D(\sim_S, N) = \frac{(N+2)(N-1)}{2} - 1$.

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

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

remaining steps, where n_1, n_2 denote numbers 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; we therefore 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 $x_1 \geq 0$ (we have to assume $n_1 \geq 1$) and $n_2 = x_2$ for $x_2 \geq 1$. We need to prove the following:

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

After elementary algebraic transformations we obtain equivalent inequalities:

$$\begin{aligned} x_1^2 + x_2^2 + 2x_1x_2 + x_1 + x_2 - 2 &\geq x_1^2 + x_2^2 + 3x_1 + x_2 - 4 \\ x_1x_2 &\geq x_1 - 1 \end{aligned}$$

Since $x_1 \geq 0$ and $x_2 \geq 1$, the last statement is obviously true. Therefore, the maximum total number of refinement steps is not larger than

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

2. $D(\sqsubseteq^{RS}, N) \geq \frac{(N+1)(N-2)}{2} + 1$: 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 with step 4':

$$4'. \text{ REMOVED}_1 := \{(p_1, p_j), (p_j, p_1) \mid j \neq 1\}; \quad \preceq_1 := \preceq_0 \setminus \text{REMOVED}_1;$$

We can proceed exactly in 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 $k = \frac{(N+1)(N-2)}{2} + 1$. □

Analogously we obtain the optimal AIP^c value for ready simulation equivalence:

$$\textbf{Theorem 7} \quad D(\sim_{RS}, N) = \frac{(N+1)(N-2)}{2}.$$

4.2 Decorated Trace Semantics

Computing preorders on regular LTSs becomes more difficult for decorated trace semantics, which are based on traces of actions possibly augmented with state properties, such as allowed/refused actions. We note that deciding trace containment is PSPACE-complete [20].

Below, we present a simple property common for all decorated trace semantics which 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 (i.e. $S \subseteq \mathbb{P} \Rightarrow \sum_{s \in S} s \in \mathbb{P}$). For any process equivalence \sim satisfying:

$$(*) \quad p \sim_{K+1} q \Leftrightarrow (p \sim_1 q) \wedge (\forall a \in A (\sum_{p': p \xrightarrow{a} p'} p' \sim_K \sum_{q': q \xrightarrow{a} q'} q'))$$

we have: $\sim_K = \sim_{K+1} \Rightarrow \forall i \in \mathbb{N} (\sim_K = \sim_{K+i})$.

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

$$\begin{aligned} & p \sim_{K+1} q \\ \Leftrightarrow & (p \sim_1 q) \wedge \forall a \in A (\sum_{p': p \xrightarrow{a} p'} p' \sim_K \sum_{q': q \xrightarrow{a} q'} q') (*) \\ \Leftrightarrow & (p \sim_1 q) \wedge \forall a \in A (\sum_{p': p \xrightarrow{a} p'} p' \sim_{K+1} \sum_{q': q \xrightarrow{a} q'} q') (\sim_K = \sim_{K+1}) \\ \Leftrightarrow & p \sim_{K+2} q (*) \quad \square \end{aligned}$$

Theorem 8 Assume a regular LTS $(\mathbb{P}, \rightarrow)$ with $|\mathbb{P}| = N$. For any process equivalence \sim being sound modulo idempotency ($x + x \sim x$), compositional w.r.t. projection, and moreover satisfying:

$$(*) \quad p \sim_{K+1} q \Leftrightarrow (p \sim_1 q) \wedge (\forall a \in A (\sum_{p': p \xrightarrow{a} p'} p' \sim_K \sum_{q': q \xrightarrow{a} q'} q'))$$

we have: $\pi_{\binom{2^N}{2}}(p) \sim \pi_{\binom{2^N}{2}}(q) \Rightarrow p \sim q$, so $D(\sim, N) \leq \binom{2^N}{2}$.

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

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 $\mathcal{L} \subseteq \text{HML}$ so that the equivalence $\sim_{\mathcal{L}}$ 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 characterization consists of finite-depth formulae. Combined with the previous result, we obtain a fairly general sufficient condition for soundness of AIP on image-finite processes. Most "strong" equivalences from 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) as well as decorated trace semantics (exponential). Constructive AIP gives an upper bound on the number of iterations required for algorithms that compute preorders on regular LTSs, e.g. [8].

Traditionally, most properties of process semantics were derived by studying each equivalence or pre-order 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. A recent research on characteristic formulae [2] is somewhat similar in spirit to ours in the sense that a common underlying structure in characteristic formulae constructions for various semantics is explored to create a general framework. The authors 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 characterizations has been given in [10]. The modal languages are more cumbersome, and other issues such as fairness in case of AIP need to be taken into account.

References

- [1] L. Aceto, B. Bloom & F. Vaandrager (1994): *Turning SOS rules into equations*. *Information and Computation* 111(1):1–52.
- [2] L. Aceto, A. Ingolfsdottir & J. Sack (2009): *Characteristic formulae for fixed-point semantics: A general framework*. In *Proc. 16th Workshop on Expressiveness in Concurrency (EXPRESS'09)*, EPTCS 8, pp. 1–15.
- [3] J.C.M. Baeten, J.A. Bergstra & J.W. Klop (1987): *On the consistency of Koomen's fair abstraction rule*. *Theoretical Computer Science* 51(1/2):129–176.
- [4] A. Barros and T. Hou (2008): *A Constructive Version of AIP Revisited*. Electronic Report PRG0802, Programming Research Group, University of Amsterdam.
- [5] J.C.M. Baeten & W.P. Weijland (1990): *Process Algebra*. *Cambridge Tracts in Theoretical Computer Science* 18, Cambridge University Press.
- [6] P. Blackburn, M. de Rijke & Y. Venema (2001): *Modal Logic*. Cambridge University Press.
- [7] B. Bloom, W.J. Fokkink & R.J. van Glabbeek (2004): *Precongruence formats for decorated trace semantics*. *ACM Transactions on Computational Logic* 5(1):26–78.
- [8] U. Celikkan & R. Cleaveland (1995): *Generating diagnostic information for behavioral preorders*. *Distributed Computing* 9(2):61–75.
- [9] R.J. van Glabbeek (1987): *Bounded nondeterminism and the approximation induction principle in process algebra*. In *Proc. 4th Symposium on Theoretical Aspects of Computer Science (STACS'87)*, LNCS 247, Springer, pp. 336–347.
- [10] R.J. van Glabbeek (1993): *The linear time–branching time spectrum II. The semantics of sequential systems with silent moves*. In *Proc. 4th Conference on Concurrency Theory (CONCUR'93)*, LNCS 715, Springer, pp. 66–81.
- [11] R.J. van Glabbeek (2001): *The linear time – branching time spectrum I; the semantics of concrete, sequential processes*. In J.A. Bergstra, A. Ponse & S.A. Smolka, editors: *Handbook of Process Algebra*, Chapter 1, Elsevier, pp. 3–99.
- [12] R.J. van Glabbeek & B. Ploeger (2008): *Correcting a space-efficient simulation algorithm*. In: *Proc. 20th Conference on Computer Aided Verification (CAV 2008)*, LNCS 5123, Springer, pp. 517–529.
- [13] R. Goldblatt (1995): *Saturation and the Hennessy–Milner property*. In A. Ponse, M. de Rijke & Y. Venema, editors: *Modal Logic and Process Algebra. A Bisimulation Perspective*, CSLI Lecture Notes 53, pp. 189–216.
- [14] M. Hennessy & R. Milner (1985): *Algebraic laws for non-determinism and concurrency*. *Journal of the ACM* 32(1):137–161.
- [15] M. Hennessy & R. Milner (1980) *On observing nondeterminism and concurrency*. In *Proc. 7th Colloquium on Automata, Languages and Programming (ICALP'80)*, LNCS 85, Springer, pp. 299–309.
- [16] M. Hollenberg (1995): *Hennessy–Milner classes and process algebra*. In A. Ponse, M. de Rijke & Y. Venema, editors: *Modal Logic and Process Algebra. A Bisimulation Perspective*, CSLI Lecture Notes 53, pp. 189–216.
- [17] K. Larsen (1990): *Proof systems for satisfiability in Hennessy–Milner logic with recursion*. *Theoretical Computer Science* 72(2&3):265–288.
- [18] S. Mauw (1987): *A constructive version of the approximation induction principle*. In *Proc. SION Conference on Computing Science in the Netherlands (CSN'87)*, pp. 235–252.
- [19] C. Stirling (2001): *Modal and Temporal Properties of Processes*. *Texts in Computer Science*, Springer.
- [20] L.J. Stockmeyer & A.R. Meyer (1973): *Word problems requiring exponential time*. In *Proc. 5th Annual ACM Symposium on Theory of Computing (STOC'73)*, ACM, pp. 1–9.