Title: **Transparent Partial Order Reduction**
Author: Stephen F. Siegel
Kind: Technical Report UD-CIS-2011/05
Notes: This is a revision of UD-CIS-2007/341

Verified Software Laboratory
Department of Computer and Information Sciences
University of Delaware
Newark DE 19716
USA
`http://vsl.cis.udel.edu`

TRANSPARENT PARTIAL ORDER REDUCTION

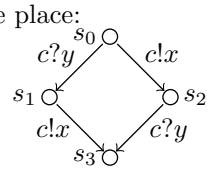
STEPHEN F. SIEGEL

Abstract. Partial Order Reduction (POR) techniques improve the basic model checking algorithm by reducing the numbers of states and transitions explored in verifying a property of the model. In the "ample set" POR framework for the verification of an LTL$_{-X}$ formula ϕ, one associates to each state s a subset T_s of the set of all transitions enabled at s. The approach requires that whenever T_s is a proper subset, the transitions in T_s must be *invisible*, i.e., their execution can never change the truth values of the atomic propositions occurring in ϕ. In this paper, we show that the invisibility restriction can be relaxed: for propositions that only occur negatively in ϕ, it suffices that the transitions in T_s merely never change the truth value from *true* to *false*, and for those that occur only positively, from *false* to *true*. This opens up opportunities for reduction, in many commonly occurring scenarios, that would not be allowed by the stricter invisibility criterion.

1. Introduction

Temporal logic model checking is a powerful tool for establishing the functional correctness of complex concurrent systems. Yet the effectiveness of model checking is often curtailed by the *state explosion problem*—the fact that the number of states of a model tends to grow exponentially in the number of concurrent processes. A variety of methods for mitigating state explosion have been proposed; among these is a family of related methods known collectively as *partial order reduction* techniques.

The basic idea behind partial order reduction is simple. An execution of a concurrent system is usually represented as an interleaved sequence of transitions from the concurrent processes. In many cases it is known *a priori* that the result of executing two transitions from distinct processes is independent of the order in which those transitions are applied. Consider, for example, a system with two processes P_1 and P_2 that access a shared channel c modeled as a FIFO queue. Suppose that only P_1 sends using c (denoted $c!x$) and only P_2 receives from c (denoted $c?y$). Then in any system state s_0 in which both a send and receive operation are enabled, the same final state s_3 will result regardless of the order in which those two operations take place:



This suggests that in searching the state space of this system, we only consider one of the two possible paths.

This material is based upon work supported by the National Science Foundation under Grant Nos. CCF-0541035, CCF-0733035, and CCF-0953210.

Of course, whether such a reduction is safe depends on the property being checked. Say, for example, we wish to verify that c is never empty; this can be expressed as the linear temporal logic (LTL) formula $\mathbf{AG}\ \neg\mathsf{empty}(c)$, where $\mathsf{empty}(c)$ is the proposition that holds precisely in those states in which c is empty. Suppose that s_0 is a state in which c contains one element and the send and receive operations are both enabled. In that case c will be empty in s_1 and so the property does not hold. However, if we were to choose to explore only the send transition from s_0, we would miss s_1 and visit only s_2 and s_3—states where c is non-empty. Our reduced search might therefore terminate without ever encountering a state in which c is empty, and we might erroneously conclude that the property holds.

For this reason, traditional POR methods, such as the *ample set* framework for verifying a next-time-free LTL formula $\mathbf{A}\phi$, impose an *invisibility* condition. A transition t is *invisible* if its execution in any state can never change the truth value of any atomic proposition occurring in ϕ. The invisibility condition requires that whenever the search is restricted to a proper subset of transitions departing from a state then all the transitions in that subset must be invisible. In our example, the only atomic proposition is $\mathsf{empty}(c)$. Since the send operation can change the value of this proposition from *true* to *false* and the receive operation can change it from *false* to *true*, neither operation is invisible, and so the search is required to explore both paths departing from s_0.

A well-known problem with this approach is that the effectiveness of the reduction technique drops off rapidly with the number of visible transitions. Several methods have been proposed to mitigate this problem (see Sec. 5). This paper contributes to those efforts by showing that the invisibility condition of the ample set framework can be safely replaced with a weaker *transparency* condition.

The notion of transparency refines that of invisibility by distinguishing between those atomic propositions that occur only *positively* in ϕ and those that occur only *negatively*. Roughly speaking, a proposition p occurs positively if some appearance of p in the syntax tree of ϕ occurs under an even number of negation operations. (In an expression of the form $p \to q$, p is considered to occur under one negation operation as the expression is equivalent to $(\neg p) \vee q$). Similarly, p occurs negatively if some appearance occurs under an odd number of negation operations. The transition t is *transparent* if for all p which occur positively in ϕ, t can never change the truth value of p from *false* to *true*, and for all p which occur negatively in ϕ, t can never change the value of p from *true* to *false*. Of course, some p may occur both positively and negatively in ϕ, in which case the transparency requirement for p reduces to the invisibility requirement for p.

In our example, the predicate $\mathsf{empty}(c)$ occurs only negatively in ϕ. Since the receive operation can never change $\mathsf{empty}(c)$ from *true* to *false*, the transparency condition permits a search in which only the receive transition departing from s_0 is followed. Note that if s_1 is a state in which c is empty then this reduced search would indeed catch the violation. On the other hand, the send operation may change $\mathsf{empty}(c)$ from *true* to *false*, and so a reduced search in which only the send operation departing from s_0 is followed would be rejected by the transparency condition.

A formal statement and proof of the transparency result are given below, but it will help at this point to provide the main intuition behind the proof. The correctness of the standard ample set approach comes down to showing that any

path through the structure resulting from the full search can be transformed to a stutter-equivalent path through the structure resulting from the reduced search. Since stutter-equivalent sequences satisfy the same next-time-free LTL path formulas, the path through the reduced structure satisfies ϕ iff the original path satisfies ϕ. This is, however, stronger than what is required for correctness: we only need to know that any path in the full structure which violates ϕ can be transformed into a reduced path that violates ϕ. Only this weaker condition holds in the transparent setting. For example, if ϕ has the form $p\mathbf{U}q$, a path violating ϕ may be transformed by stuttering *and also* by changing any number of values taken on by p and q from *true* to *false*. It is not hard to see that such a transformed sequence must also violate $p\mathbf{U}q$.

The remainder of this paper is organized as follows. The formal framework and statement of the main result are presented in Sec. 2. The proof of the main result is then given in Sec. 3. Some applications and preliminary experiments applying the transparent technique to the verification of message-passing programs are outlined in Sec. 4. Related work is discussed in Sec. 5 and conclusions and future work are discussed in Sec. 6.

2. The Main Theorem

2.1. **Transparent transitions.** We adopt the notation of [2, Chap. 10]. Let AP be a set of *atomic propositions*, and $\mathcal{M} = (S, T, S_0, L)$ a *state transition system* over AP. This means that S is a finite set of *states*, $S_0 \subseteq S$ is the set of *initial states*, T is a finite set of *transitions*, i.e., if $\alpha \in T$ then $\alpha \subseteq S \times S$, and $L : S \to 2^{AP}$ is a labeling function. We assume that there are elements $\mathsf{true}, \mathsf{false} \in AP$ with the property that $\mathsf{true} \in L(s)$ and $\mathsf{false} \notin L(s)$ for all $s \in S$. For $s \in S$ we let $\mathsf{enabled}(s) = \{\alpha \in T \mid \exists s' \in S\colon (s, s') \in \alpha\}$. For $\alpha \in T$ we let $\mathsf{enabled}(\alpha) = \{s \in S \mid \exists s' \in S\colon (s, s') \in \alpha\}$. We assume all transitions are *deterministic*, that is, if $s \in \mathsf{enabled}(\alpha)$ then there is a unique $s' \in S$ for which $(s, s') \in \alpha$; we denote this state s' by $\alpha(s)$.

Definition 2.1. *Given $\omega = (P, N) \in 2^{AP} \times 2^{AP}$, let $\sqsubseteq_\omega$ denote the binary relation on 2^{AP} defined by $S \sqsubseteq_\omega T \Leftrightarrow (S \cap P \subseteq T \wedge S \supseteq T \cap N)$. We say α* is ω-transparent *if for all $s \in \mathsf{enabled}(\alpha)$, $L(\alpha(s)) \sqsubseteq_\omega L(s)$.*

Hence if α is ω-transparent, then α always preserves the falsity of propositions in P, and the truth of propositions in N. In particular, α is invisible iff α is (AP, AP)-transparent. Note that $\sqsubseteq_\omega$ is a preorder: it is reflexive and transitive, but not necessarily antisymmetric.

2.2. **Formulas.** Recall that an LTL$_{-X}$ formula is a state formula of the form $\mathbf{A}\phi$, where ϕ is a path formula over AP such that the operators used in ϕ all lie in the set $\{\neg, \rightarrow, \wedge, \vee, \mathbf{G}, \mathbf{F}, \mathbf{U}, \mathbf{W}, \mathbf{R}\}$. Let $\mathbf{A}\phi$ be an LTL$_{-X}$ formula over AP. Consider the syntax tree associated to ϕ. To each node u in this tree is associated a subformula ϕ_u of ϕ. If u is a leaf node then $\phi_u \in AP$. We define, inductively, a number $\mathrm{sgn}(u) \in \{-1, 1\}$ for each node u. If u is the root node, let $\mathrm{sgn}(u) = 1$. Now assume u is any node and we have defined $\mathrm{sgn}(u)$. If u corresponds to any operation other than $\neg$ or $\rightarrow$, then we let $\mathrm{sgn}(v) = \mathrm{sgn}(u)$ for all children v of u. If u corresponds to $\neg$, then we let $\mathrm{sgn}(v) = -\mathrm{sgn}(u)$ for the sole child v of u. If u corresponds to $\rightarrow$, then we let $\mathrm{sgn}(v) = -\mathrm{sgn}(u)$ and $\mathrm{sgn}(w) = \mathrm{sgn}(u)$, where v and w are respectively the left and right children of u.

Definition 2.2. *Let* $\omega = (P, N) \in 2^{AP} \times 2^{AP}$. *An* ω-path formula *is a path formula* ϕ *such that for all leaf nodes* u *in the syntax tree of* ϕ, *if* $\operatorname{sgn}(u) = 1$ *then* $\phi_u \in P$ *and if* $\operatorname{sgn}(u) = -1$ *then* $\phi_u \in N$.

2.3. **Reduced Space.** The POR framework requires two additional structures: an *independence relation*, and a choice of *ample sets*.

Definition 2.3. *An* independence relation $I \subseteq T \times T$ *is a symmetric, antireflexive relation on* T *such that, for any* $(\alpha, \beta) \in I$, *and for all* $s \in S$ *for which* $\alpha, \beta \in \mathsf{enabled}(s)$, *the following both hold: (i)* $\alpha \in \mathsf{enabled}(\beta(s))$, *and (ii)* $\alpha(\beta(s)) = \beta(\alpha(s))$. *We say that* α *and* β *are* independent *if* $(\alpha, \beta) \in I$. *We say* α *and* β *are* dependent *if* $(\alpha, \beta) \notin I$.

A Kripke structure $M = (S, R, S_0, L)$ may be obtained from $\mathcal{M}$ by defining R so that $(s, s') \in R$ iff $(s, s') \in \alpha$ for some $\alpha \in T$. A *path in* M is an ordered pair $\pi = \langle s, \zeta \rangle$, where $s \in S$ and $\zeta = \alpha_0 \alpha_1 \ldots$ is a (finite or infinite) sequence of transitions, such that there exist $s_0, s_1, \ldots \in S$ satisfying $s_0 = s$ and $(s_i, s_{i+1}) \in \alpha_i$ for all i. It follows from the deterministic hypothesis that if the s_i exist, they are unique.

Fix $\omega \in 2^{AP} \times 2^{AP}$. Suppose for each $s \in S$, we are given a set $\mathsf{ample}(s) \subseteq \mathsf{enabled}(s)$. These define the *reduced Kripke structure* $M^\flat = (S, R^\flat, S_0, L)$, where $(s, s') \in R^\flat$ iff there exists $\alpha \in \mathsf{ample}(s)$ such that $\alpha(s) = s'$. A path in $M^\flat$ is a path in M for which $\alpha_i \in \mathsf{ample}(s_i)$ for all i. We repeat here the four hypotheses on ample sets from [2, Chap. 10], the only difference being we have replaced "invisible" with "ω-transparent" in **C2**.

- **C0** For all $s \in S$, $\mathsf{ample}(s) = \emptyset \Leftrightarrow \mathsf{enabled}(s) = \emptyset$.
- **C1** For all $s \in S$, along every path in M that starts at s, a transition that is dependent on a transition in $\mathsf{ample}(s)$ cannot occur without a transition in $\mathsf{ample}(s)$ occurring first.
- **C2**$_\omega$ For all $s \in S$, if $\mathsf{ample}(s) \neq \mathsf{enabled}(s)$ then every $\alpha \in \mathsf{ample}(s)$ is ω-transparent.
- **C3** There is no cycle in $M^\flat$ containing a state at which some transition α is enabled, but is never included in $\mathsf{ample}(s)$ for any state s in the cycle.

Theorem 2.4. *Let* AP *be a set of atomic propositions,* $\mathcal{M} = (S, T, S_0, L)$ *a state transition system over* AP, $\omega \in 2^{AP} \times 2^{AP}$, *and* I *an independence relation for* $\mathcal{M}$. *Suppose we are given, for each* $s \in S$, *a set* $\mathsf{ample}(s) \subseteq \mathsf{enabled}(s)$, *such that* **C0**, **C1**, **C2**$_\omega$, and **C3** *all hold. Let* M *be the Kripke structure corresponding to* $\mathcal{M}$, *and* $M^\flat$ *the reduced Kripke structure. Then for any* ω*-path formula* ϕ, $M \models \mathbf{A}\phi \Leftrightarrow M^\flat \models \mathbf{A}\phi$.

3. Proof

In this section, we prove Thm. 2.4. Hence we assume we are given a state transition system $\mathcal{M} = (S, T, S_0, L)$ over a set of atomic propositions AP, $\omega \in 2^{AP} \times 2^{AP}$, an ω-path formula ϕ, an independence relation I for $\mathcal{M}$, and ample sets $\mathsf{ample}(s)$ ($s \in S$) satisfying **C0**, **C1**, **C2**$_\omega$, and **C3**. As before, we let M be the Kripke structure corresponding to $\mathcal{M}$, and $M^\flat$ the reduced Kripke structure.

3.1. **Preliminaries.** Throughout, we will write "transparent" for "ω-transparent".

Let $\mathbf{N} = \{0, 1, \ldots.\}$ and $\mathbf{N}^\bullet = \mathbf{N} \cup \{\infty\}$. For any sequence ζ we define $|\zeta| \in \mathbf{N}^\bullet$ to be the length of ζ.

Let $\pi = \langle s, \zeta \rangle$ be a path in M. Say $\zeta = \beta_0\beta_1 \dots$. Define $\mathsf{state}_0(\pi) = s$ and $\mathsf{state}_{i+1}(\pi) = \beta_i(\mathsf{state}_i(\pi))$ for $i \geq 0$. Let $\mathsf{first}(\pi) = \mathsf{state}_0(\pi)$. The *$i$-th suffix of π* is the path

$$\mathsf{S}_i(\pi) = \langle \mathsf{state}_i(\pi), \beta_i\beta_{i+1} \dots \rangle.$$

The *length of π*, denoted $|\pi|$, is defined to be $|\zeta|$. If $|\pi| = n < \infty$, $\mathsf{last}(\pi)$ is defined to be $\mathsf{state}_n(\pi)$. If π is finite and σ is any path with $\mathsf{first}(\sigma) = \mathsf{last}(\pi)$, then we define $\pi * \sigma$ to be the concatenation of π with σ; it is a path starting from $\mathsf{first}(\pi)$.

We now define certain transformations on paths in M. The purpose of these is to move a single transition ahead of independent transitions, a standard technique used to reason about POR (cf. [2,12]).

Definition 3.1. *Let $\pi = \langle s, \beta_0\beta_1 \dots \rangle$ be an infinite path in M, $i \in \mathbf{N}^\bullet$, $\alpha \in T$, and suppose all of the following hold:*

(a) $\alpha \in \mathsf{enabled}(s)$,
(b) *if $i < \infty$ then $\alpha = \beta_i$,*
(c) *for all $j < i$, α is independent of β_j, and*
(d) *if $i > 0$ then α is transparent.*

Then we define

$$\Gamma_i^\alpha \pi = \begin{cases} \langle \alpha(s), \beta_0 \dots \beta_{i-1}\beta_{i+1} \dots \rangle & \text{if } i < \infty \\ \langle \alpha(s), \beta_0\beta_1 \dots \rangle & \text{if } i = \infty. \end{cases}$$

Hence Γ_i^α is a function from a certain subset of the set of all infinite paths in M, to the set of infinite paths in M. The fact that $\Gamma_i^\alpha \pi$ is a path follows easily from the first three conditions of Def. 3.1, and Def. 2.3.

For $i < \infty$, we have moved transition $\alpha = \beta_i$ to the left, past i independent transitions, and then taken the suffix S_1 of the result. For $i = \infty$, we have inserted the independent transition α at the beginning and then taken S_1 of the result.

We now establish notation for performing a sequence of transformations of the kind described above. Let $j \geq 0$, $\zeta = \alpha_0 \dots \alpha_{j-1}$ a sequence of length j of elements of T, and $\nu = (i_0, \dots, i_{j-1})$ a sequence of length j of non-negative integers. We define a function Γ_ν^ζ, which again is defined on a subset of infinite paths in M. If $j = 0$ (i.e., ν and ζ are empty sequences) we let $\Gamma_\nu^\zeta \pi = \pi$ for all infinite paths π. For $j \geq 1$, we let

$$\Gamma_\nu^\zeta = \Gamma_{i_{j-1}}^{\alpha_{j-1}} \cdots \Gamma_{i_0}^{\alpha_0}, \tag{1}$$

where the product denotes function composition. Implicit in (1) is the fact that $\Gamma_\nu^\zeta \pi$ is defined iff $\Gamma_{i_0}^{\alpha_0} \pi$ is defined and $\Gamma_{i_1}^{\alpha_1} \Gamma_{i_0}^{\alpha_0} \pi$ is defined, and so on.

Lemma 3.2. *If $\Gamma_\nu^\zeta \pi$ is defined then $\sigma = \langle \mathsf{first}(\pi), \zeta \rangle$ is a finite path in M and $\mathsf{last}(\sigma) = \mathsf{first}(\Gamma_\nu^\zeta \pi)$.*

Proof. The proof proceeds by induction on $|\zeta|$. For $|\zeta| = 0$, σ is the path of length 0 starting at $\mathsf{first}(\pi)$, and so $\mathsf{last}(\sigma) = \mathsf{first}(\pi) = \mathsf{first}(\Gamma_\nu^\zeta \pi)$, as required.

Assume $j \geq 0$ and the Lemma holds whenever $|\zeta| \leq j$. Suppose now that $\zeta = \alpha_0 \dots \alpha_j$, $\nu = (i_0, \dots, i_j)$, and $\Gamma_\nu^\zeta \pi$ is defined. Let $\zeta' = \alpha_0 \dots \alpha_{j-1}$ and $\nu' = (i_0, \dots, i_{j-1})$. Since $\Gamma_\nu^\zeta \pi = \Gamma_{i_j}^{\alpha_j} \Gamma_{\nu'}^{\zeta'} \pi$ is defined, $\pi' = \Gamma_{\nu'}^{\zeta'} \pi$ is defined. Hence by the inductive hypothesis, $\langle s, \zeta' \rangle$ is a path, where $s = \mathsf{first}(\pi)$, and $\mathsf{last}\langle s, \zeta' \rangle = \mathsf{first}(\pi')$.

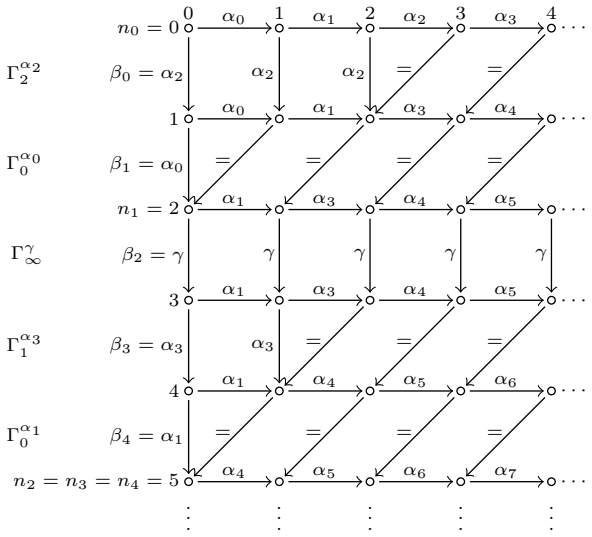


FIGURE 1. A transformation $\rho \rightsquigarrow \sigma$. The top row represents ρ; the leftmost column represents σ.

Now since $\Gamma_{i_j}^{\alpha_j} \pi'$ is defined, $\alpha_j \in \mathsf{enabled}(\mathsf{first}(\pi')) = \mathsf{enabled}(\mathsf{last}\langle s, \zeta' \rangle)$, which means that $\langle s, \zeta \rangle$ is a path. Moreover,

$$\mathsf{last}\langle s, \zeta \rangle = \alpha_j(\mathsf{last}\langle s, \zeta' \rangle) = \alpha_j(\mathsf{first}(\pi')) = \mathsf{first}(\Gamma_{i_j}^{\alpha_j} \pi') = \mathsf{first}(\Gamma_{\nu}^{\zeta} \pi),$$

completing the inductive step. □

We now define a relation on infinite paths in M.

Definition 3.3. *Let $\rho = \langle s, \alpha_0 \alpha_1 \dots \rangle$ and $\sigma = \langle s, \beta_0 \beta_1 \dots \rangle$ be infinite paths in M starting at the same state s. Write $\rho \rightsquigarrow \sigma$ to mean there exist $i_0, i_1, \dots \in \mathbf{N}^\bullet$ such that (i) $i_j = 0$ for an infinite number of j, and (ii) $\Gamma_{i_0,\dots,i_{j-1}}^{\beta_0,\dots,\beta_{j-1}} \rho$ is defined for all $j \geq 0$.*

An example illustrating the concepts introduced so far is given in Fig. 1. In the figure, each node represents a state. The rows are numbered starting with the top (row 0), and working down (rows $1, 2, \dots$). The columns are numbered similarly from left to right. Row 0 represents a path $\rho = \langle s, \alpha_0 \alpha_1 \dots \rangle$, while the leftmost column represents a path $\sigma = \langle s, \beta_0 \beta_1 \dots \rangle$ for which $\rho \rightsquigarrow \sigma$. The transformation from ρ to σ is illustrated one step at a time. Hence, the path in row 1 represents $\Gamma_2^{\alpha_2} \rho$, the path in row 2 represents $\Gamma_0^{\alpha_0} \Gamma_2^{\alpha_2} \rho$, and so on. An edge denotes either a transition from one state to another, or equality, i.e., that the source and destination states are equal. From the figure, we can discern that α_2, α_3, and γ must all be transparent transitions, that α_2 is independent of α_0 and α_1, and so on. The numbers $n_0, n_1, \dots$ will be defined in Sec. 3.2.

The proof of Thm. 2.4 proceeds by showing that any path in the full state space which does not satisfy an ω-path formula ϕ can be transformed into a path in the reduced space which does not satisfy ϕ. The following lemma deals with the transformation: it states that a path ρ in the full space can be transformed into a path σ in the reduced space so that $\rho \rightsquigarrow \sigma$. The transformation proceeds by moving

ample transitions forward when this is possible or inserting new ample transitions when it is not. All four properties **C0**–**C3** are used in the proof.

Lemma 3.4. *If ρ is an infinite path in M starting at a state s then there exists an infinite path σ in $M^\flat$ starting at s such that $\rho \rightsquigarrow \sigma$.*

Proof. First, we define elements $\beta_j \in T$ and $i_j \in \mathbf{N}^\bullet$ for all $j \geq 0$, by induction on j. Along the way, we will show that for all j

$$\pi_j = \Gamma_{i_0,\ldots,i_{j-1}}^{\beta_0,\ldots,\beta_{j-1}} \rho \text{ is defined, and } \langle s, \beta_0 \ldots \beta_{j-1}\rangle \text{ is a path in } M^\flat. \tag{2}$$

This will imply that $\sigma = \langle s, \beta_0\beta_1 \ldots\rangle$ is a path in $M^\flat$ and it will only remain to show that an infinite number of the i_j are 0 to conclude that $\rho \rightsquigarrow \sigma$.

The case $j = 0$ is vacuous, so suppose $j \geq 0$ and that the β_k and i_k have been defined for $0 \leq k < j$ to satisfy (2). Write $\pi_j = \langle s_j, \gamma_{j,0}\gamma_{j,1} \ldots\rangle$. If $\gamma_{j,0} \in \mathsf{ample}(s_j)$, let $i_j = 0$ and $\beta_j = \gamma_{j,0}$. It is immediate that $\Gamma_{i_j}^{\beta_j} \pi_j$ is defined, and thus, by Lem. 3.2, $\chi = \langle s, \beta_0 \ldots \beta_j\rangle$ is a path. By the inductive hypothesis, $\langle s, \beta_0, \ldots, \beta_{j-1}\rangle$ is a path in $M^\flat$, so since $\beta_j \in \mathsf{ample}(s_j)$, χ is a path in $M^\flat$.

So assume $\gamma_{j,0} \notin \mathsf{ample}(s_j)$. Then $\gamma_{j,0} \in \mathsf{enabled}(s_j) \setminus \mathsf{ample}(s_j)$, and so by $\mathbf{C2}_\omega$, all of the transitions in $\mathsf{ample}(s_j)$ are transparent.

Now either there is some $k \geq 1$ such that $\gamma_{j,k} \in \mathsf{ample}(s_j)$, or there is no such k.

If the first is the case, choose the least such k. Then by **C1**, $\gamma_{j,k}$ is independent of $\gamma_{j,l}$ for all $l < k$, so we may take $i_j = k$ and $\beta_j = \gamma_{j,k}$. Since all transitions in $\mathsf{ample}(s_j)$ are transparent, it is again the case that $\Gamma_{i_j}^{\beta_j} \pi_j$ is defined, and we may reason exactly as before to see that $\langle s, \beta_0, \ldots, \beta_j\rangle$ is a path in $M^\flat$.

So suppose there is no such k. Then **C1** implies that for all $k \geq 0$, $\gamma_{j,k}$ is independent of every transition in $\mathsf{ample}(s_j)$. By **C0**, $\mathsf{ample}(s_j)$ is nonempty. So we let $i_j = \infty$ and β_j be any element of $\mathsf{ample}(s_j)$, completing the inductive step.

It remains to see that $i_j = 0$ for an infinite number of j. Suppose that this is not the case. Then there exists $j \geq 0$ such that for all $k \geq j$, $i_k > 0$. Hence $\gamma_{k,0} = \gamma_{j,0}$ for all $k \geq j$. By construction, this implies $\gamma_{j,0} \in \mathsf{enabled}(s_k) \setminus \mathsf{ample}(s_k)$ for $k \geq j$. Now $s_k = \mathsf{state}_k(\sigma)$ for all k, and, since S is finite, there must exist $l > k \geq j$ such that $s_l = s_k$. Hence s_k lies on a cycle in $M^\flat$ in which $\gamma_{j,0}$ is enabled, but not included in the ample set for any of the states of the cycle, contradicting **C3**. □

The following is an easy exercise in the definitions (recall from Sec. 3.1 that S_i denotes the i^{th} suffix of a sequence):

Lemma 3.5. *Suppose π is an infinite path in M, $i \in \mathbf{N}$, $j \in \mathbf{N}^\bullet$, and $\Gamma_j^\alpha \pi$ is defined. Then*

$$\mathsf{S}_i \Gamma_j^\alpha \pi = \begin{cases} \Gamma_{j-i}^\alpha \mathsf{S}_i\pi & \text{if } j \geq i \\ \mathsf{S}_{i+1}\pi & \text{if } j \leq i. \end{cases}$$

3.2. **A Proposition.** We have seen that any path ρ in M can be transformed into a path σ in M' such that $\rho \rightsquigarrow \sigma$. Our goal is to show that $\rho \not\models \phi \Rightarrow \sigma \not\models \phi$ for any ω-path formula ϕ. To do this we must first examine the relation between ρ and σ in greater detail. We will see there is a non-decreasing sequence of integers $(n_d)_{d\geq 0}$ such that the d-th state s_d in ρ corresponds to the n_d-th state t_{n_d} in σ, in the following sense. First, there is a path τ_d of transparent transitions from s_d to t_{n_d}. Furthermore, all of the transitions in σ except the last from t_{n_d} to $t_{n_{d+1}}$ are transparent. By the transitivity of $\sqsubseteq_\omega$, this shows that any $p \in P$ that does not

hold at s_d also does not hold at t_i for $n_d \leq i < n_{d+1}$, and that any $p \in N$ which does hold at s_d will also hold at the t_i. Furthermore, the two paths departing from s_d—the suffix of ρ and the path formed by first taking τ_d and then the suffix of σ—are related by $\rightsquigarrow$. This is made precise in the following:

Proposition 3.6. *Let $\rho = \langle s, \alpha_0\alpha_1\dots\rangle$ and $\sigma = \langle s, \beta_0\beta_1\dots\rangle$ be infinite paths in M starting at the same state s and suppose $\rho \rightsquigarrow \sigma$. Then there exist finite paths $\tau_0, \tau_1, \dots$ in M, and non-negative integers $n_0, n_1, \dots$ such that the following hold for all $d \geq 0$:*

(a) $d \leq n_d \leq n_{d+1}$,
(b) τ_d *consists entirely of transparent transitions,*
(c) $\mathsf{first}(\tau_d) = \mathsf{state}_d(\rho)$ *and* $\mathsf{last}(\tau_d) = \mathsf{state}_{n_d}(\sigma)$,
(d) $\mathsf{S}_d(\rho) \rightsquigarrow \tau_d * \mathsf{S}_{n_d}(\sigma)$, *and*
(e) β_k *is transparent whenever* $n_d \leq k < n_{d+1} - 1$.

The remainder of this section will be devoted to a proof of Prop. 3.6.

By Def. 3.3, there are non-negative integers $i_0, i_1, \dots$ such that (i) $i_k = 0$ for an infinite number of k, and (ii) for all $k \geq 0$, $\Gamma_{i_0,\dots,i_{k-1}}^{\beta_0,\dots,\beta_{k-1}} \rho$ is defined.

Fix $d \geq 0$. Define non-negative integers $m_{d,k}$ $(k \geq 0)$ as follows:

$$m_{d,0} = d, \qquad m_{d,k+1} = \begin{cases} m_{d,k} & \text{if } i_k \geq m_{d,k} \\ m_{d,k} - 1 & \text{otherwise.} \end{cases} \tag{3}$$

Hence $m_{d,0}, m_{d,1}, \dots$ is a non-increasing sequence of integers starting at d. At each step, the sequence either decreases by one or remains unchanged. If it reaches 0, it remains at 0 from that point forward, since every $i_k \geq 0$. We claim that in fact the sequence must reach 0: this follows easily from the fact that there are an infinite number of k for which $i_k = 0$. We let n_d be the least k for which $m_{d,k} = 0$.

In Fig. 1, we may interpret the $m_{d,k}$ as follows: consider the path that begins at the state in column d of row 0, and that progresses by following the unique edge that moves down one row at each step. Then $m_{d,k}$ is the column number of the state that results after taking k steps in this path. For example, for $d = 2$, we have $m_{2,0} = m_{2,1} = 2$, $m_{2,2} = m_{2,3} = m_{2,4} = 1$, and $m_{2,5} = 0$. In particular, $n_2 = 5$.

The following gathers together some facts about the $m_{d,k}$:

Lemma 3.7. *The following hold for all $d, k \geq 0$:*

(a) $m_{d,k} - 1 \leq m_{d,k+1} \leq m_{d,k}$,
(b) $m_{d,n_d} = 0 \ \wedge \ (d \geq 1 \Rightarrow m_{d,n_d-1} > 0)$,
(c) $m_{d,k+1} = m_{d,k} \Leftrightarrow i_k \geq m_{d,k}$,
(d) $m_{d,k} \geq d - k$,
(e) $|\{l \mid 0 \leq l < n_d, i_l \geq m_{d,l}\}| = n_d - d$,
(f) $m_{d,k} = m_{d+1,k} \Rightarrow m_{d,k+1} = m_{d+1,k+1}$, *and*
(g) $m_{d,k} \leq m_{d+1,k} \leq m_{d,k} + 1$.

Proof. Statements (a)–(c) are immediate from the definitions. To prove (d), fix d, and use induction on k: for $k = 0$, the statement holds since $m_{d,0} = d$, and the inductive step follows from (a).

To see (e), let $A = \{0, 1, \dots, n_d - 1\}$ and $B = \{l \in A \mid i_l < m_{d,l}\}$. Let $\bar{m}_{d,l} = m_{d,l} - m_{d,l+1}$ $(l \in A)$. By (a), $m_{d,l} \in \{0, 1\}$ for all $l \in A$. Moreover,

$\bar{m}_{d,l} = 1 \Leftrightarrow l \in B$, by (c). By (b), $m_{d,n_d} = 0$, and hence

$$d = m_{d,0} = \sum_{l \in A} \bar{m}_{d,l} = |\{l \in A \mid \bar{m}_{d,l} = 1\}| = |B|.$$

So $|A \setminus B| = |A| - |B| = n_d - d$, proving (e).

Now (f) holds since, by (3), the value of $m_{d,k+1}$ depends only on $m_{d,k}$ and i_k.

We now turn to the proof of (g). We first claim that for all $d, k \geq 0$,

$$m_{d,k} \leq m_{d+1,k}. \tag{4}$$

To show this, we fix d, and use induction on k. For $k = 0$, (4) reduces to the statement $d \leq d + 1$, which clearly holds. Suppose now that (4) holds and we wish to show it still holds when k is replaced by $k + 1$. There are two cases to consider: either (i) $m_{d,k} < m_{d+1,k}$, or (ii) $m_{d,k} = m_{d+1,k}$. In the first case, we have

$$m_{d,k+1} \leq m_{d,k} \leq m_{d+1,k} - 1 \leq m_{d+1,k+1},$$

as required. In the second case, we have $m_{d,k+1} = m_{d+1,k+1}$, by (f), which completes the inductive step.

We now show that for $d, k \geq 0$,

$$m_{d+1,k} - m_{d,k} \leq 1, \tag{5}$$

which, in light of (4), will complete the proof of (g). Again fix $d \geq 0$ and use induction on k. For $k = 0$, (5) reduces to the statement $d + 1 - d \leq 1$. Suppose now that (5) holds, and we wish to show that it holds with $k + 1$ in place of k. Then $m_{d+1,k} - m_{d,k}$ must equal either 0 or 1. In the first case, we have $m_{d+1,k} = m_{d,k}$, and so by (f), $m_{d+1,k+1} - m_{d,k+1} = 0$, and the inductive step holds.

So assume that $m_{d+1,k} - m_{d,k} = 1$. There are again two cases to consider: either (i) $m_{d,k+1} = m_{d,k}$ or (ii) $m_{d,k+1} = m_{d,k} - 1$. If (i) is the case, we have

$$m_{d+1,k+1} - m_{d,k+1} \leq m_{d+1,k} - m_{d,k} = 1,$$

and the inductive step holds. If (ii) is the case, then $i_k < m_{d,k}$, by (c). Hence by (4), $i_k < m_{d,k} \leq m_{d+1,k}$, and again by (c), $m_{d+1,k+1} = m_{d+1,k} - 1$. Hence

$$m_{d+1,k+1} - m_{d,k} = m_{d+1,k} - 1 - m_{d,k} + 1 = m_{d+1,k} - m_{d,k} \leq 1,$$

completing the inductive step and the proof of Lem. 3.7. □

We now return to the proof of Prop. 3.6. According to Lem. 3.7(a), if $m_{d+1,k} = 0$ then $m_{d,k} \leq m_{d+1,k} = 0$. Hence $n_d \leq n_{d+1}$. Moreover, Lem. 3.7(d) implies $m_{d,d-1} \geq 1$, which shows that $n_d \geq d$, establishing Prop. 3.6(a).

Suppose $n_d \leq k < n_{d+1} - 1$. Then $m_{d,k} = 0$ but $m_{d+1,k} \neq 0$, which implies $m_{d+1,k} = 1$ by Lem. 3.7(g). We also have $m_{d+1,k+1} = 1$, since $k + 1 < n_{d+1}$. Hence $m_{d+1,k+1} = m_{d+1,k}$, which, by Lem. 3.7(c), implies $i_k \geq m_{d+1,k} = 1$. By Def. 3.1, this means that β_k is transparent, proving Prop. 3.6(e).

Fix $d \geq 0$. For each $k \geq 0$, we define a sequence $\xi_{d,k}$ of transitions and a sequence $\mu_{d,k}$ of non-negative integers. For $k = 0$, these are both the empty sequence. Assuming they have been defined for k, we let

$$\xi_{d,k+1} = \begin{cases} \xi_{d,k} * (\beta_k) & \text{if } i_k \geq m_{d,k} \\ \xi_{d,k} & \text{otherwise} \end{cases}, \quad \mu_{d,k+1} = \begin{cases} \mu_{d,k} * (i_k - m_{d,k}) & \text{if } i_k \geq m_{d,k} \\ \mu_{d,k} & \text{otherwise.} \end{cases}$$

By Lem. 3.7(e), we have $|\xi_{d,n_d}| = |\mu_{d,n_d}| = n_d - d$. Now if $k < n_d$, then $m_{d,k} > 0$, so if $i_k \geq m_{d,k}$ then $i_k > 0$ and therefore β_k is transparent. Hence ξ_{d,n_d} consists solely of transparent transitions.

On the other hand, if $k \geq n_d$ then $m_{d,k} = 0$ and so $i_k \geq m_{d,k}$. It follows that

$$\xi_{d,k} = \xi_{d,n_d} * \beta_{n_d}\beta_{n_d+1}\dots\beta_{k-1} \quad \text{and} \quad \mu_{d,k} = \mu_{d,n_d} * (i_{n_d}, i_{n_d+1}, \dots, i_{k-1}) \tag{6}$$

for $k \geq n_d$. For any $k \geq 0$, let $\zeta_k = \beta_0 \dots \beta_{k-1}$ and let $\nu_k = i_0 \dots i_{k-1}$. Recall that, by assumption, $\Gamma_{\nu_k}^{\zeta_k}\,\rho$ is defined for all k.

Lemma 3.8. *For all $k \geq 0$, $\Gamma_{\mu_{d,k}}^{\xi_{d,k}}\,\mathsf{S}_d\rho = \mathsf{S}_{m_{d,k}}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho$.*

Proof. Implicit in Lem. 3.8 is the claim that the left-hand side is defined. We prove the lemma by induction on k. For $k = 0$, the statement reduces to the equation $\mathsf{S}_d(\rho) = \mathsf{S}_d(\rho)$. Now suppose that the statement holds at k and we wish to show that it holds at $k+1$. There are two cases to consider: (i) $i_k \geq m_{d,k}$ and (ii) $i_k < m_{d,k}$.

Suppose $i_k \geq m_{d,k}$, so $m_{d,k+1} = m_{d,k}$. Since

$$\Gamma_{i_k}^{\beta_k}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho = \Gamma_{\nu_{k+1}}^{\zeta_{k+1}}\,\rho \tag{7}$$

is defined, Lem. 3.5 implies

$$\mathsf{S}_{m_{d,k}}\,\Gamma_{i_k}^{\beta_k}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho = \Gamma_{i_k-m_{d,k}}^{\beta_k}\,\mathsf{S}_{m_{d,k}}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho. \tag{8}$$

In particular, the right hand side of (8) is defined. Hence

$$\Gamma_{\mu_{d,k+1}}^{\xi_{d,k+1}}\,\mathsf{S}_d\rho = \Gamma_{i_k-m_{d,k}}^{\beta_k}\,\Gamma_{\mu_{d,k}}^{\xi_{d,k}}\,\mathsf{S}_d\rho = \Gamma_{i_k-m_{d,k}}^{\beta_k}\,\mathsf{S}_{m_{d,k}}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho \tag{9}$$

is defined. Moreover, combining (7)–(9), we have

$$\Gamma_{\mu_{d,k+1}}^{\xi_{d,k+1}}\,\mathsf{S}_d\rho = \mathsf{S}_{m_{d,k}}\,\Gamma_{i_k}^{\beta_k}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho = \mathsf{S}_{m_{d,k}}\,\Gamma_{\nu_{k+1}}^{\zeta_{k+1}}\,\rho = \mathsf{S}_{m_{d,k+1}}\,\Gamma_{\nu_{k+1}}^{\zeta_{k+1}}\,\rho,$$

completing the inductive step for this case.

Suppose instead that $i_k < m_{d,k}$. Then $i_k \leq m_{d,k+1} = m_{d,k} - 1$. Hence $\xi_{d,k+1} = \xi_{d,k}$ and $\mu_{d,k+1} = \mu_{d,k}$. Moreover, Lem. 3.5 implies

$$\mathsf{S}_{m_{d,k}-1}\,\Gamma_{i_k}^{\beta_k}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho = \mathsf{S}_{m_{d,k}}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho. \tag{10}$$

Whence

$$\Gamma_{\mu_{d,k+1}}^{\xi_{d,k+1}}\,\mathsf{S}_d\rho = \Gamma_{\mu_{d,k}}^{\xi_{d,k}}\,\mathsf{S}_d\rho = \mathsf{S}_{m_{d,k}}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho = \mathsf{S}_{m_{d,k}-1}\,\Gamma_{i_k}^{\beta_k}\,\Gamma_{\nu_k}^{\zeta_k}\,\rho = \mathsf{S}_{m_{d,k+1}}\,\Gamma_{\nu_{k+1}}^{\zeta_{k+1}}\,\rho,$$

completing the inductive step for this case as well. □

For $d > 0$, we let $\tau_d = \langle \mathsf{state}_d(\rho), \xi_{d,n_d}\rangle$. From Lemmas 3.2 and 3.8, we conclude that τ_d is a path, and, since $m_{d,n_d} = 0$,

$$\begin{aligned}\mathsf{last}(\tau_d) &= \mathsf{first}\,\Gamma_{\mu_{d,n_d}}^{\xi_{d,n_d}}\,\mathsf{S}_d\rho = \mathsf{first}\,\mathsf{S}_0\,\Gamma_{\nu_{n_d}}^{\zeta_{n_d}}\,\rho \\ &= \mathsf{first}\,\Gamma_{\nu_{n_d}}^{\zeta_{n_d}}\,\rho = \mathsf{last}\langle s, \beta_0 \dots \beta_{n_d-1}\rangle = \mathsf{state}_{n_d}(\sigma).\end{aligned}$$

This proves Prop. 3.6(c), and also shows that $\tau_d * \mathsf{S}_{n_d}(\sigma)$ is a path. We have already seen τ_d consists solely of transparent transpositions, proving Prop. 3.6(b).

Write $\tau_d * \mathsf{S}_{n_d}(\sigma) = \langle \mathsf{state}_d(\rho), \gamma_0\gamma_1\dots\rangle$. By (6),

$$\gamma_0 \dots \gamma_{k-1} = \xi_{d,k+d} \qquad (k \geq n_d - d). \tag{11}$$

Write $\mu_{d,n_d} = (i'_0, i'_1, \ldots, i'_{n_d-d-1})$, and define $i'_k = i_{k+d}$ $(k \geq n_d - d)$. It follows from (6) that $(i'_0, \ldots, i'_{k-1}) = \mu_{d,k+d}$ for $k \geq n_d - d$. Since $i_k = 0$ for an infinite number of k, $i'_k = 0$ for an infinite number of k. Moreover, for all $k \geq 0$,

$$\Gamma^{\gamma_0,\ldots,\gamma_{k-1}}_{i'_0,\ldots,i'_{k-1}} \mathsf{S}_d\rho = \Gamma^{\xi_{d,k+d}}_{\mu_{d,k+d}} \mathsf{S}_d\rho$$

is defined, by Lem. 3.8. Hence $\mathsf{S}_d(\rho) \rightsquigarrow \tau_d * \mathsf{S}_{n_d}(\sigma)$, proving Prop. 3.6(d). This completes the proof of Prop. 3.6.

3.3. **Formula Preservation.** In this section, we prove the following:

Proposition 3.9. *If $\rho \rightsquigarrow \sigma$ and $\rho \not\models \phi$ then $\sigma \not\models \phi$.*

We first show that it suffices to prove Prop. 3.9 for ω-path formulas in which the only operators are $\mathbf{U}$, $\neg$, and $\wedge$. For suppose we have done this, and that we are given an ω-path formula ϕ. We can convert ϕ to a path formula ψ involving only the three operators using the identities $\theta \vee \chi \equiv \neg((\neg\theta) \wedge (\neg\chi))$, $\theta \rightarrow \chi \equiv (\neg\theta) \vee \chi$, $\mathbf{F}\theta \equiv \mathsf{true}\mathbf{U}\theta$, $\mathbf{G}\theta \equiv \neg\mathbf{F}\neg\theta$, $\theta\mathbf{R}\chi \equiv \neg((\neg\theta)\mathbf{U}(\neg\chi))$, and $\theta\mathbf{W}\chi \equiv (\mathbf{G}\theta) \vee (\theta\mathbf{U}\chi)$. Because these identities are "parity-preserving," it is not hard to see that ψ is also an ω-path formula. Moreover, we have $M \models \mathbf{A}\phi \Leftrightarrow M \models \mathbf{A}\psi$, and $M^\flat \models \mathbf{A}\phi \Leftrightarrow M^\flat \models \mathbf{A}\psi$, since for any infinite path π in M, $\pi \models \phi \Leftrightarrow \pi \models \psi$. By assumption, Prop. 3.9 holds for ψ, i.e., $M \models \mathbf{A}\psi \Leftrightarrow M^\flat \models \mathbf{A}\psi$. Hence $M \models \mathbf{A}\phi \Leftrightarrow M^\flat \models \mathbf{A}\phi$, as required.

So assume the only operators occurring in ϕ are $\mathbf{U}$, $\neg$, and $\wedge$. The proof will work by induction over the syntax tree for ϕ, beginning at the leaf nodes and ending at the root. For each node u, we define two statements $\mathfrak{p}_u$ and $\mathfrak{q}_u$. In the following, "$\models_u$" stands for "$\models$" if $\mathrm{sgn}(u) = -1$, or "$\not\models$" if $\mathrm{sgn}(u) = 1$.

($\mathfrak{p}_u$) for all infinite paths ρ, σ in M for which $\rho \rightsquigarrow \sigma$, $\rho \models_u \phi_u \Rightarrow \sigma \models_u \phi_u$

($\mathfrak{q}_u$) for all infinite paths $\pi = \langle s, \alpha_0\alpha_1 \ldots \rangle$ in M for which α_0 is transparent, $\pi \models_u \phi_u \Rightarrow \mathsf{S}_1(\pi) \models_u \phi_u$.

We will show by induction that $\mathfrak{p}_u \wedge \mathfrak{q}_u$ holds for all u. This will complete the proof of Prop. 3.9, since if u is the root node then $\mathfrak{p}_u$ is just a restatement of Prop. 3.9. But first, we will need the following:

Lemma 3.10. *Suppose that u is a node in the syntax tree of ϕ, $\mathfrak{p}_u \wedge \mathfrak{q}_u$ holds, ρ and σ are infinite paths in M for which $\rho \rightsquigarrow \sigma$, $d \geq 0$, and n_d is as in Prop. 3.6. Then the following hold:*

(a) *If $\mathsf{S}_d(\rho) \models_u \phi_u$ then $\mathsf{S}_{n_d}(\sigma) \models_u \phi_u$.*
(b) *If $\mathsf{S}_i(\rho) \models_u \phi_u$ for all i such that $0 \leq i < d$, then $\mathsf{S}_k(\sigma) \models_u \phi_u$ for all k such that $0 \leq k < n_d$.*

Proof. Let $n_0, n_1, \ldots$, and $\tau_0, \tau_1, \ldots$, be as in Prop. 3.6. We assume $\mathrm{sgn}(u) = -1$, the case $\mathrm{sgn}(u) = 1$ being entirely similar.

We first prove (a). By Prop. 3.6(d), $\mathsf{S}_d(\rho) \rightsquigarrow \tau_d * \mathsf{S}_{n_d}(\sigma)$. By $\mathfrak{p}_u$, this means that $\tau_d * \mathsf{S}_{n_d}(\sigma) \models \phi_u$. By Prop. 3.6(b), τ_d consists entirely of transparent transitions. By repeated application of $\mathfrak{q}_u$, we conclude that $\mathsf{S}_{n_d}(\sigma) \models \phi_u$, as required.

We now turn to (b). By (a), $\mathsf{S}_{n_i}(\sigma) \models \phi_u$ for $0 \leq i < d$. Now suppose $0 \leq k < n_d$. By Prop. 3.6(a), $0 = n_0 \leq n_1 \leq \ldots$, and so $k = n_i + j$ for some $0 \leq i < d$ and $0 \leq j < n_{i+1}$. Write $\sigma = \langle s, \beta_0\beta_1 \ldots \rangle$. By Prop. 3.6(e), $\beta_{n_i}, \beta_{n_i+1}, \ldots, \beta_{n_i+j-1}$ are all transparent, and so repeated applications of $\mathfrak{q}_u$ imply that $\mathsf{S}_k(\sigma) \models \phi_u$. □

We now return to the proof of Prop. 3.9. Suppose that u is a leaf node, so $p = \phi_u \in AP$. Then for any infinite path ρ in M, $\rho \models \phi_u \Leftrightarrow p \in L(\mathsf{first}(\rho))$. Since $\rho \rightsquigarrow \sigma \Rightarrow \mathsf{first}(\rho) = \mathsf{first}(\sigma)$, $\mathfrak{p}_u$ holds. Statement $\mathfrak{q}_u$ follows from Def. 2.1 since $\mathsf{S}_1(\pi) \models \phi_u \Leftrightarrow p \in L(\alpha_0(s))$.

Now suppose u is any node and $\mathfrak{p}_v \wedge \mathfrak{q}_v$ holds for all children v of u.

Suppose first that u has two children, v and w, and that $\phi_u = \phi_v \wedge \phi_w$. Then $\operatorname{sgn}(v) = \operatorname{sgn}(w) = \operatorname{sgn}(u)$. So $\mathfrak{p}_u \wedge \mathfrak{q}_u$ holds since, for any infinite path ρ, $\rho \models \phi_u$ iff $\rho \models \phi_v$ and $\rho \models \phi_w$.

Suppose instead that u has a single child v, and that $\phi_u = \neg\phi_v$. Then $\operatorname{sgn}(v) = -\operatorname{sgn}(u)$. Now $\mathfrak{p}_u \wedge \mathfrak{q}_u$ holds since, for any infinite path ρ, $\rho \models \phi_u$ iff $\rho \not\models \phi_v$.

Suppose now that u has two children, v and w, and that $\phi_u = \phi_v \mathbf{U} \phi_w$. We will first consider the case $\operatorname{sgn}(u) = -1$. In this case, $\operatorname{sgn}(v) = \operatorname{sgn}(w) = -1$.

Let us first prove $\mathfrak{p}_u$. So assume ρ, σ are infinite paths starting at a state s, and $\rho \rightsquigarrow \sigma$. Suppose $\rho \models \phi_u$. We must show that $\sigma \models \phi_u$. Since $\rho \models \phi_v \mathbf{U} \phi_w$, there exists $d \geq 0$ such that $\mathsf{S}_d(\rho) \models \phi_w$ and, for all $0 \leq i < d$, $\mathsf{S}_i(\rho) \models \phi_v$. By Lem. 3.10(a), $\mathsf{S}_{n_d}(\sigma) \models \phi_w$, while by Lem. 3.10(b), $\mathsf{S}_k(\sigma) \models \phi_v$ whenever $0 \leq k < n_d$. This shows that $\sigma \models \phi_v \mathbf{U} \phi_w$, as required.

Let us now prove $\mathfrak{q}_u$. So suppose $\pi = \langle s, \alpha_0 \alpha_1 \ldots \rangle$ is an infinite path in M for which α_0 is transparent, and that $\pi \models \phi_u$. We must show that $\mathsf{S}_1(\pi) \models \phi_u$. Since $\phi_u = \phi_v \mathbf{U} \phi_w$, there exists $d \geq 0$ such that $\mathsf{S}_d(\pi) \models \phi_w$ and $\mathsf{S}_i(\pi) \models \phi_v$ for $0 \leq i < d$. If $d > 0$, then we have

$$\mathsf{S}_{d-1}\mathsf{S}_1(\pi) = \mathsf{S}_d(\pi) \models \phi_w \quad \text{and} \quad \mathsf{S}_j\mathsf{S}_1(\pi) = \mathsf{S}_{j+1}(\pi) \models \phi_v \qquad (0 \leq j < d-1)$$

and hence $\mathsf{S}_1(\pi) \models \phi_u$, as required. On the other hand, if $d = 0$, then $\pi \models \phi_w$, and applying the inductive hypothesis $\mathfrak{q}_w$ we conclude that $\mathsf{S}_1(\pi) \models \phi_w$, which implies $\mathsf{S}_1(\pi) \models \phi_u$. This completes the proof that $\mathfrak{p}_u \wedge \mathfrak{q}_u$ holds if $\operatorname{sgn}(u) = -1$.

Consider now the case $\operatorname{sgn}(u) = 1$. In this case, $\operatorname{sgn}(v) = \operatorname{sgn}(w) = 1$. We first prove $\mathfrak{p}_u$. So suppose that $\rho \not\models \phi_v \mathbf{U} \phi_w$. There are two possibilities: (i) for all $d \geq 0$, $\mathsf{S}_d(\rho) \not\models \phi_w$, or (ii) for some $d \geq 0$, $\mathsf{S}_d(\rho) \not\models \phi_v$ and $\mathsf{S}_i(\rho) \not\models \phi_w$ for $0 \leq i \leq d$.

If (i) is the case, then by Lem. 3.10(b), for all $d \geq 0$ and $k < n_d$, $\mathsf{S}_k(\sigma) \not\models \phi_w$. However, by Prop. 3.6(a), $\lim_{d\to\infty} n_d = \infty$, hence $\mathsf{S}_k(\sigma) \not\models \phi_w$ for all $k \geq 0$, which means that $\sigma \not\models \phi_v \mathbf{U} \phi_w$, as required.

If (ii) is the case, then it follows from Lem. 3.10(a) that $\mathsf{S}_{n_d}(\sigma) \not\models \phi_v$, while Lem. 3.10(b) implies that $\mathsf{S}_k(\sigma) \not\models \phi_w$ for $0 \leq k < n_d$. Again, this means that $\sigma \not\models \phi_v \mathbf{U} \phi_w$, establishing $\mathfrak{p}_u$.

We now turn to the proof of $\mathfrak{q}_u$ in the case that $\operatorname{sgn}(u) = 1$. So suppose $\pi = \langle s, \alpha_0 \alpha_1 \ldots \rangle$ is an infinite path in M for which α_0 is transparent, and that $\pi \not\models \phi_u$. We must show that $\mathsf{S}_1(\pi) \not\models \phi_u$. Again, there are two cases to consider: (i) for all $d \geq 0$, $\mathsf{S}_d(\pi) \not\models \phi_w$, or (ii) for some $d \geq 0$, $\mathsf{S}_d(\pi) \not\models \phi_v$ and $\mathsf{S}_i(\pi) \not\models \phi_w$ for $0 \leq i \leq d$.

If (i) is the case, then we certainly have $\mathsf{S}_d\mathsf{S}_1(\pi) = \mathsf{S}_{d+1}(\pi) \not\models \phi_w$ (for all $d \geq 0$), which implies $\mathsf{S}_1(\pi) \not\models \phi_u$, as required.

So suppose (ii) is the case. If $d > 0$, then

$$\mathsf{S}_{d-1}\mathsf{S}_1(\pi) = \mathsf{S}_d(\pi) \not\models \phi_v \quad \text{and} \quad \mathsf{S}_j\mathsf{S}_1(\pi) = \mathsf{S}_{j+1}(\pi) \not\models \phi_w \ \ (0 \leq j \leq d-1),$$

whence again $\mathsf{S}_1(\pi) \not\models \phi_u$, as required. So suppose $d = 0$. Then $\pi \not\models \phi_v$ and $\pi \not\models \phi_w$. By the inductive hypotheses $\mathfrak{q}_v$ and $\mathfrak{q}_w$, this means that $\mathsf{S}_1(\pi) \not\models \phi_v$ and $\mathsf{S}_1(\pi) \not\models \phi_w$. Hence $\mathsf{S}_1(\pi) \not\models \phi_v \mathbf{U} \phi_w = \phi_u$. This establishes $\mathfrak{q}_u$ for the case $\operatorname{sgn}(u) = 1$, and completes the proof of Prop. 3.9.

3.4. **Conclusion.** We can now complete the proof of Thm. 2.4. One direction is clear: since any path in $M^\flat$ is a path in M, we have $M \models \mathbf{A}\phi \Rightarrow M^\flat \models \mathbf{A}\phi$. So we must show that $M \not\models \mathbf{A}\phi \Rightarrow M^\flat \not\models \mathbf{A}\phi$. So suppose $M \not\models \mathbf{A}\phi$, i.e., there is some infinite path ρ in M, whose start state s is in S_0, such that $\rho \not\models \phi$. By Lem. 3.4, there exists an infinite path σ in $M^\flat$ starting at s such that $\rho \rightsquigarrow \sigma$. By Prop. 3.9, $\sigma \not\models \phi$. Hence $M^\flat \not\models \mathbf{A}\phi$, as required.

4. Evaluation

4.1. **TMC.** To evaluate the effectiveness of transparent partial order reduction, I developed a simple Java tool called the Transparent Model Checker (TMC). The source code for TMC and all artifacts for the experiments reported here are available at `http://vsl.cis.udel.edu`. The TMC input language encodes a parametrized, concurrent system which may contain shared variables and/or asynchronous message-passing primitives. There is one type, integer, and the standard integer operations and comparators are supported. A non-zero value is interpreted as *true* and 0 as *false*, as in C. Each process comprises a set of locations, and, at each location, a set of clauses, each ending with the new location to which to move after executing that clause. There are four different kinds of locations. A *choice* location can have any number of clauses, each with its own guard expression; when control is at such a location, the guards are evaluated and one that evaluates to *true* is chosen nondeterministically. The remaining kinds all have one clause. The clause at a *send* location is a send statement; that at a *receive* location is a receive statement. Finally, the clause for an *assignment* location has an arbitrary guard and an atomic assignment statement.

The `send` statement takes three arguments: an expression which is evaluated to obtain the message value, the PID of the destination process, and a message tag. The arguments for `recv` are the variable into which the message should be stored (or `null` if the message value is to be ignored), the PID of the sending process (or `null` if a message from any process may be received), and the message tag (or `null` if a message with any tag may be received). A receive statement with a `null` source argument is known as a *wildcard* receive and is translated as n distinct transitions, all of which are identical receive statements, except for the source process, which varies from 0 to $n-1$. There is an implicit FIFO channel for each ordered pair of processes, with a fixed capacity (specified at verification time). A send is enabled when the channel is not full, the receive when it is not empty.

A TMC program defines a transition system with the usual interleaving semantics. A simple message-passing example is given in Fig. 2. Based on [1, Fig. 3.12], the program encodes a coordinator barrier, a standard barrier implementation in which one process, the coordinator, keeps track of the number of processes that have entered the barrier. Each worker process p sends a message to the coordinator when it enters the barrier, and then waits for the coordinator to send a reply signifying that p may leave the barrier. The coordinator receives a message from each worker, and then signals each worker to leave, and repeats. The code uses a parameter N (all parameters have integer type), and indicates that one coordinator process and N worker processes should be instantiated.

Following the process definitions is a sequence of predicate and formula definitions. The first defines the elements of AP. These may also be parametrized. For example, for $1 \leq i \leq N$, predicate `atStart(i)` holds precisely when worker $i-1$

is at location `loc0`. Formulas are LTL_{-X} formulas over AP. Formula `p1` specifies that if a process enters the barrier it will remain in the barrier until all processes are in the barrier. Formula `p2` specifies that every process will be outside of the barrier infinitely often, while `p3` states that all processes will be simultaneously in the barrier infinitely often.

A shared-variable example is given in Fig. 3. This example, based on [1, Fig. 4.3], encodes a producer-consumer system with M producers and N consumers. It uses two variables `empty` and `full` as binary semaphores to signal when a buffer is empty and full, respectively. The formula `read` states that every message produced is eventually consumed.

Finally, a TMC program may contain one or more `check` commands to verify a formula over some range of parameter values, using one of three possible reduction schemes: full (no reduction), invisible (ample-set POR based on the standard invisibility criterion), and transparent POR using the relaxed transparent criterion. For each parameter tuple, TMC (1) constructs the (reduced) state space, (2) invokes Java PathFinder's LTL to Büchi converter [4] to construct a Büchi automaton corresponding to the negation of the formula, and (3) performs a nested depth-first search on the product automaton to determine whether the intersection of the language of the state space and the property automaton is empty.

Note TMC does not use "on-the-fly" reduction, which combines the computation of the reduced space and the product automaton, as there are questions about the soundness of that technique even for the standard invisible-based POR (see Sec. 5).

4.2. **Reduction.** The reduced structure is determined as follows. First, two transitions are considered independent if they are from distinct processes and do not involve shared variables. Ample sets are chosen during the search: given the current state and a process p, the set of all enabled transitions in p is a candidate ample set if none of these transitions involves shared variables and, if p is at a receive location, every receiving channel has at least one queued message. A candidate set is rejected if (a) it is empty, (b) it contains a transition that is not transparent, or (c) it contains a transition that leads to a state already on the search stack. If no candidate set survives rejection, the set of all enabled transitions is used for the ample set, otherwise one candidate set is chosen using some heuristic.

This strategy ensures that if a proper ample set T consists of transitions from a single process p, then any transition dependent on a transition in T is also in p; furthermore none of these dependent transitions are enabled and no action from another process can enable them. This suffices to show that **C1** is satisfied. It is not hard to see the other ample set conditions are satisfied as well.

Our heuristic proceeds by first iterating through the processes in order of increasing PID, searching for a proper ample set that consists entirely of *invisible* local, non-send transitions. If none is found, it then attempts to find a set consisting of invisible send transitions. If this fails, it then repeats these attempts but allowing transparent transitions (that are not necessarily invisible). If this also fails then the set of all enabled transitions is used. The choice of this heuristic is based on experience which suggests that invisible ample sets give the best reduction (when they exist), and those consisting of receives or local operations generally do better than those consisting of sends.

```
model coordBarrier(N);
proc Coordinator [1] {
  int i;
  loc0: i=1; goto loc1;
  loc1: when (i>N) goto loc4;
        when (!(i>N)) goto loc2;
  loc2: recv(null,i,0); goto loc3;
  loc3: i=i+1; goto loc1;
  loc4: i=1; goto loc5;
  loc5: when (i>N) goto loc0;
        when (!(i>N)) goto loc6;
  loc6: send(0,i,0); goto loc7;
  loc7: i=i+1; goto loc5;
}
```

```
proc Worker [N] {
  loc0: send(0,0,0); goto loc1;
  loc1: recv(null,0,0); goto loc0;
}
predicate atStart(i) =
     Worker[i-1]@loc0;
formula in(i) = !atStart(i);
predicate enter(i) = nempty(i,0,0);
formula allIn = and{i=1..N} in(i);
formula p1 = and{i=1..N}
    [](enter(i) -> in(i) U allIn);
formula p2 = and{i=1..N} []<>!in(i);
formula p3 = []<>and{i=1..N}in(i);
```

FIGURE 2. TMC source for Coordinator Barrier

```
model ProducerConsumer(M,N);
int empty = 1; int full = 0; int buffer = 0;
proc Producer[M] {
  int data = 0;
  loc0: when (1) goto loc1; when (1) goto loc2;
  loc1: data = 0; goto loc3;
  loc2: data = 1; goto loc3;
  loc3: when (empty>0) empty = empty - 1; goto loc4;
  loc4: buffer = data; goto loc5;
  loc5: full = full + 1; goto loc0;
}
proc Consumer[N] {
  int result = 0;
  loc0: when (full>0) full = full - 1; goto loc1;
  loc1: result = buffer; goto loc2;
  loc2: empty = empty + 1; goto loc3;
  loc3: result = 0; goto loc0;
}
predicate produce1(i) = Producer[i].data && Producer[i]@loc4;
predicate consume1(j) = Consumer[j].result;
formula read = and{i=0..M-1}[](produce1(i) -> <>or{j=0..N-1}consume1(j));
check read for chanSize=0, M=2, N=1..10 using transparent;
```

FIGURE 3. TMC source code for shared memory Producer-Consumer

Transparency of transitions is determined statically, once a TMC program and sets of negative and positive predicates are given. The analysis is based on recognizing certain patterns, such as linear expressions. For example, given a predicate of the from $x \geq C$, where x is a program variable and C a constant, and an assignment statement of the form $x = x + D$, where D is a non-negative constant, the analysis will conclude that the statement preserves the truth of the predicate.

4.3. **Experiments.** Experiments were run on 4 scalable TMC programs, each with one or more LTL formulas. For each formula, TMC was executed in full, invisible, and transparent mode, scaling until the number of transitions in the (reduced)

structure exceeded 10^6. The number of transitions were recorded in each case, and graphs summarizing this data are given in Fig. 4.

The first program is the coordinator barrier system shown in Fig. 2 and described above. The results for property `p1` are shown in graph (a) of Fig. 4. The graphs for the other two properties result in only minor changes to the invisible and transparent graphs; the full graph is of course identical since it is independent of the property.

The second program derives from the "dissemination barrier" system described in [1, Sec. 3.4.3]. Instead of using a coordinator, the n processes use a symmetric protocol to impose the barrier among themselves. The protocol proceeds in stages $0, 1, \ldots, \lceil \log_2(n) \rceil - 1 \equiv m$. In stage j, for each i $(0 \leq i < n)$, process i sends a message to process $i + 2^j$ and receives a message from process $i - 2^j$ using a send-receive call. (Process IDs are reduced modulo n.) The same three properties used for the coordinator barrier were used here, though the atomic propositions are defined slightly differently: enter_i holds when process i is at the state s_1 which is the target of the transition labeled by local event λ; in_i holds when process i is at neither the start state nor at s_1. Graph (b) gives the results for the third property; again, the other two are very similar.

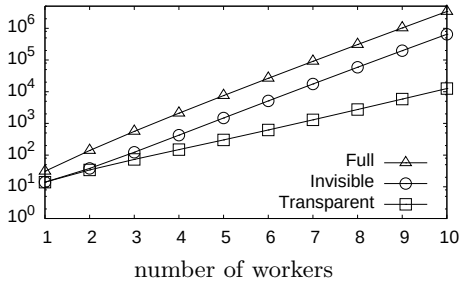
The third program is derived from the "multiple producer, single consumer" (MPSC) program of [15, Ex. 2.18]. In this message-passing program, n producers send messages to a single consumer, which consumes from the producers in a cyclic fashion. The data is abstracted away altogether. The property states that no producer becomes permanently blocked, which can be expressed in LTL as $\mathbf{A} \bigwedge_i \mathbf{GF} \neg \mathsf{full}(c_i)$, where c_i is the channel used by producer i to send to the consumer. All transitions other than the send and receive statements are invisible, while all transitions other than the receive statements are transparent. To scale, the channel size is fixed at 3 and the number of producers is increased. The results are shown in graph (c).

The fourth program is the shared-variable producer-consumer example shown in Fig. 3 and described above.

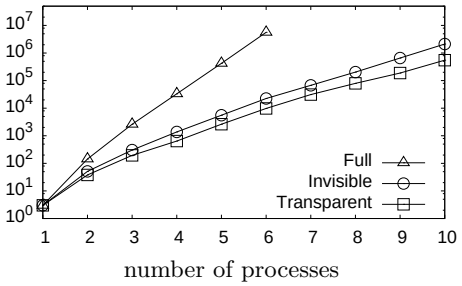
In all cases, the number of transitions executed in the full search increased exponentially as the system scaled. The effect of the two reduction strategies is more varied. In (a), all three functions are exponential; invisibility reduces the number of transitions by a constant factor; transparency actually reduces the exponent, yielding a 517x reduction from invisible for 10 workers. In (b), invisibility yields a very significant savings over full, and transparency makes a small improvement over that. The most dramatic effect of transparency is seen in (c), where transparency reduces the exponential functions for full and invisible to a linear function. In (d), all functions are still exponential, the but transparency again makes a dramatic improvement over invisibility, allowing the system to scale further while remaining within the bound of 10^6 transitions.

5. Related Work

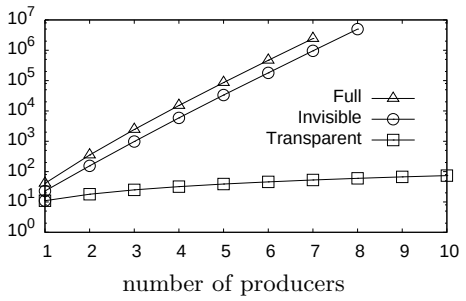
The reader is referred to [5], [10], [2, Chap. 10], and the references cited in those works for a guide to the large literature on partial order reduction. Two of the pioneering works on LTL_{-X}-preserving POR methods are [16] and [8], both of which state the invisibility condition.



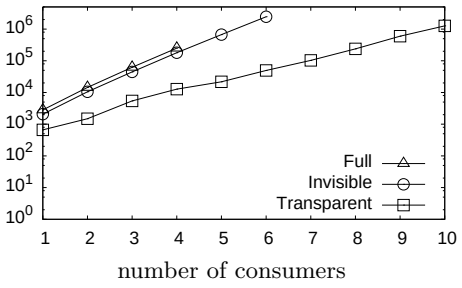
(a) Coordinator Barrier, $\mathbf{A} \bigwedge_i \mathbf{G}(\mathsf{enter}_i \rightarrow \mathsf{in}_i \mathbf{U} \bigwedge_i \mathsf{in}_i)$



(b) Dissemination Barrier, $\mathbf{AGF}\neg\mathsf{in}_0$



(c) Multiple Producer Single Consumer, chansize = 3, $\mathbf{A} \bigwedge_i \mathbf{GF}\neg\mathsf{full}(c_i)$



(d) Producer-Consumer, 2 producers, $\mathbf{A} \bigwedge_i \mathbf{G}(\mathsf{produce}_i \rightarrow \mathbf{F} \bigvee_j \mathsf{consume}_j)$

FIGURE 4. Number of transitions (y-axis) in Kripke structure using no reduction (full), invisible, and transparent reduction.

A POR-like algorithm for verifying properties such as freedom from potential deadlock in models of parallel programs that use the Message Passing Interface is studied in [13]. Freedom from potential deadlock can be expressed as $\mathbf{AG}\neg\mathsf{phalt}$, where phalt holds in any state for which the only enabled transitions are sends which cannot be immediately followed by their matching receives. The key ingredients in that algorithm are (1) the introduction of synchronous transitions and (2) the observation that receives, synchronous, and local event transitions cannot change the truth value of phalt from *true* to *false*. In the language of this paper, those transitions are transparent to $\mathbf{G}\neg\mathsf{phalt}$. In fact, the main theorem of this paper evolved out of an attempt to generalize the observations of [13].

A relaxation of the invisibility condition in the context of "on-the-fly" POR is investigated in [11]. The idea is to dynamically reduce the set of invisible transitions as the search progresses. The algorithm requires a specific construction for the Büchi automaton which annotates states with subformulas of $\neg\phi$. The relaxed condition requires that a transition be invisible only with respect to the propositions occurring in the subformulas associated to the property component of the current state of the search. This approach appears to be orthogonal to, and compatible with, the transparent relaxation. However, the technique depends on [9], about which some open questions remain. As explained in [14], the proof of the main

theorem of [9] contains a gap; I am not aware of a counterexample to the theorem itself, but I also do not see how to plug the gap.

Other relaxed-visibility techniques are described in [7]. Those techniques apply to two specific CTL formulas (**EF** p and **AG EF** p) and are investigated in the context of Petri nets. They involve certain sets of transitions called *up sets* and *down sets*. Though the definitions are somewhat complex, the core idea is the distinction between transitions that can only change the truth value of p from *true* to *false* and those that can only change that value from *false* to *true*, an idea that is also central in this paper.

There have been several investigations into the relaxation of other POR conditions. In [3], for example, a dynamic notion of independence is used in the context of multi-threaded programs manipulating a shared heap; the question of whether two transitions are independent is a function of the current state in the search. It is likely that many of these approaches are independent of the invisibility criterion, and may be safely combined with the transparent technique, though this will require careful study.

Theorem 5.12 of [12] asserts that the "*forming path*" [12, Def. 5.6] relation from the reduced sub-state space to the full state space is a "*visible simulation*" [12, Def. 5.5]. That result would provide a much simpler proof of the main theorem of this paper. However, [12, Thm. 5.12] is incorrect; a counterexample is given in [14].

6. Conclusion

We have described a simple modification to the ample set POR framework. The modification is a relaxation of the invisibility condition that distinguishes between propositions that occur only positively in the formula being checked, and those that occur only negatively. The modified framework may open up opportunities for reduction that do not exist in the standard framework. Furthermore, any heuristic for choosing ample sets in the traditional framework can be extended so that the modified algorithm does no worse (in terms of numbers of states or transitions explored) than the standard algorithm.

To take advantage of the modified framework, one must be able to identify program statements that preserve the truth (or falsity) of propositions occurring in the formula. While sophisticated automated reasoning approaches might be brought to bear on this problem, there are plenty of commonly-occurring scenarios that can be easily (and probably automatically) detected. For example, if c is a FIFO channel, then a send operation on c preserves the truth of $\mathsf{full}(c)$ and the falsity of $\mathsf{empty}(c)$; a receive on c preserves the truth of $\mathsf{empty}(c)$ and the falsity of $\mathsf{full}(c)$. If x is a numeric variable then the assignment $x \leftarrow x-1$ preserves the truth of $x \leq N$ and $x \leftarrow x+1$ preserves the truth of $x \geq N$ (ignoring overflows). If p is a predicate that holds iff the flow of control of a process is at a particular point then any statement that results in transferring control to that point preserves the truth of p, and any statement that results in transferring control to any other point preserves the falsity of p.

We have applied the improved algorithm in the context of the verification of some simple message-passing and shared-variable concurrent programs. Our experiments show various degrees of (and in some cases, dramatic) improvement over the standard algorithm. The examples are relatively simple, and a broader study using more complex examples is needed in order to ascertain the effectiveness of the

transparent framework. Beyond this, the most important work remaining involves combining the optimization described here with other reduction techniques, which will require careful analysis and further studies.

References

[1] G. R. Andrews. *Foundations of Multithreaded, Parallel, and Distributed Programming*. Addison-Wesley, 2000.

[2] E. M. Clarke, Jr., O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, Cambridge, 1999.

[3] M. B. Dwyer, J. Hatcliff, Robby, and V. P. Ranganath. Exploiting object escape and locking information in partial-order reductions for concurrent object-oriented programs. *Form. Methods Syst. Des.*, 25:199–240, September 2004.

[4] D. Giannakopoulou and F. Lerda. From states to transitions: Improving translation of LTL formulae to Büchi automata. In D. Peled and M. Vardi, editors, *Formal Techniques for Networked and Distributed Sytems – FORTE 2002*, volume 2529 of *LNCS*, pages 308–326. Springer, 2002.

[5] P. Godefroid. *Partial-Order Methods for the Verification of Concurrent Systems: An Approach to the State-Explosion Problem*. Springer, Berlin, 1996.

[6] G. J. Holzmann and D. Peled. An improvement in formal verification. In D. Hogrefe and S. Leue, editors, *Proceedings of the 7th IFIP WG6.1 Intl. Conference on Formal Description Techniques (Forte '94)*, volume 6 of *IFIP Conference Proceedings*, pages 197–211. Chapman & Hall, 1995.

[7] L. M. Kristensen, K. Schmidt, and A. Valmari. Question-guided stubborn set methods for state properties. *Formal Methods in System Design*, 29(3):215–251, Nov. 2006.

[8] D. Peled. All from one, one for all: On model checking using representatives. In C. Courcoubetis, editor, *Computer-Aided Verification, 5th Intl. Conference (CAV '93)*, volume 697 of *LNCS*, pages 409–423. Springer-Verlag, 1993.

[9] D. Peled. Combining partial order reductions with on-the-fly model-checking. *Formal Methods in System Design*, 8(1):39–64, Jan. 1996.

[10] D. Peled. Ten years of partial order reduction. In A. J. Hu and M. Y. Vardi, editors, *Computer Aided Verification, 10th Intl. Conference (CAV '98)*, volume 1427 of *LNCS*, pages 17–28. Springer, 1998.

[11] D. Peled, A. Valmari, and I. Kokkarinen. Relaxed visibility enhances partial order reduction. *Formal Methods in System Design*, 19(3):275–289, Nov. 2001.

[12] D. A. Peled. Partial order reduction: linear and branching temporal logics and process algebras. In D. A. Peled, V. R. Pratt, and G. J. Holzmann, editors, *Partial Order Methods in Verification: Workshop on Partial Order Methods in Verification, July 24–26, 1996, Princeton University*, volume 29 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 233–258. American Mathematical Society, 1997.

[13] S. F. Siegel. Efficient verification of halting properties for MPI programs with wildcard receives. In R. Cousot, editor, *Verification, Model Checking, and Abstract Interpretation: 6th International Conference, VMCAI 2005, Paris, January 17–19, 2005, Proceedings*, volume 3385 of *LNCS*, pages 413–429, 2005.

[14] S. F. Siegel. Reexamining two results in partial order reduction. Technical Report UD-CIS-2011/06, `http://vsl.cis.udel.edu/pubs/por_tr_2011.html`. University of Delaware, 2011.

[15] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI—The Complete Reference, Volume 1: The MPI Core*. MIT Press, second edition, 1998.

[16] A. Valmari. A stubborn attack on state explosion. *Formal Methods in System Design*, 1(4):297–322, Dec. 1992.

The Verified Software Laboratory, Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716, USA

E-mail address: `siegel@cis.udel.edu`