Measuring Progress of Probabilistic LTL Model Checking

Elise Cormie-Bowins* and Franck van Breugel†

DisCoVeri Group, Department of Computer Science, York University,
4700 Keele Street, Toronto, ON, M3J 1P3, Canada

Recently, Zhang and Van Breugel introduced the notion of a progress measure for a probabilistic model checker. Given a linear-time property ϕ and a description of the part of the system that has already been checked, the progress measure returns a real number in the unit interval. The real number captures how much progress the model checker has made towards verifying ϕ. If the progress is zero, no progress has been made. If it is one, the model checker is done. They showed that the progress measure provides a lower bound for the measure of the set of execution paths that satisfy ϕ. They also presented an algorithm to compute the progress measure when ϕ is an invariant.

In this paper, we present an algorithm to compute the progress measure when ϕ is a formula of a positive fragment of linear temporal logic. In this fragment, we can express invariants but also many other interesting properties. The algorithm is exponential in the size of ϕ and polynomial in the size of that part of the system that has already been checked. We also present an algorithm to compute a lower bound for the progress measure in polynomial time.

1   Introduction

Due to the infamous state space explosion problem, model checking a property of source code that contains randomization often fails. In many cases, the probabilistic model checker simply runs out of memory without reporting any useful information. In [11], Zhang and Van Breugel proposed a progress measure for probabilistic model checkers. This measure captures the amount of progress the model checker has made with its verification effort. Even if the model checker runs out of memory, the amount of progress may provide useful information.

Our aim is to develop a theory that is applicable to probabilistic model checkers in general. Our initial development has been guided by a probabilistic extension of the model checker Java PathFinder (JPF) [9]. This model checker can check properties, expressed in linear temporal logic (LTL), of Java code containing probabilistic choices.

We model the code under verification as a probabilistic transition system (PTS), and the systematic search of the system by the model checker as the set of explored transitions of the PTS. We focus on linear-time properties, in particular those expressed in LTL. The progress measure is defined in terms of the set of explored transitions and the linear-time property under verification. The progress measure returns a real number in the interval $[0, 1]$. The larger this number, the more progress the model checker has made with its verification effort.

Zhang and Van Breugel showed that their progress measure provides a lower bound for the measure of the set of execution paths that satisfy the linear-time property under verification. If, for example, the progress is 0.9999, then the probability that we encounter a violation of the linear-time property when we run the code is at most 0.0001. Hence, despite the fact the model checker may fail by running out of memory, the verification effort may still be a success by providing an acceptable upper bound on the probability of a violation of the property.

The two main contributions of this paper are

1. a characterization of the progress measure for a positive fragment of LTL. This fragment includes invariants, and most examples found in, for example, [2, Section 5.1] can be expressed in this fragment. This characterization forms the basis for an algorithm to compute the progress measure.

2. a polynomial time algorithm to compute a lower bound for the progress measure for the positive fragment of LTL. The lower bound is tight for invariants, that is, this algorithm computes the progress for invariants.

2   A Progress Measure

In this section, we review some of the key notions and results of [11]. We represent the system to be verified by the probabilistic model checker as a probabilistic transition system.

Definition 1 *A probabilistic transition system is a tuple* $\langle S, T, AP, s_0, \text{source}, \text{target}, \text{prob}, \text{label} \rangle$ *consisting of*

- *a countable set S of states,*

- *a countable set T of transitions,*

- *a set AP of atomic propositions,*

- *an initial state* s_0,

- *a function* $\text{source} : T \to S$,

- *a function* $\text{target} : T \to S$,

- *a function* $\text{prob} : T \to (0, 1]$, *and*

- *a function* $\text{label} : S \to 2^{AP}$

such that

- $s_0 \in S$ *and*

- *for all* $s \in S$, $\sum \{ \text{prob}(t) \mid \text{source}(t) = s \} = 1$.

Example 2 *The probabilistic transition system* $\mathscr{S}$ *depicted by*
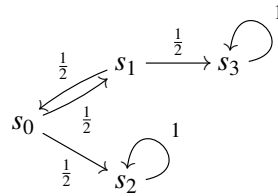
has three states and six transitions. In this example, we use the indices of the source and target to name the transitions. For example, the transition from s_0 *to* s_2 *is named* t_{02}. *Given this naming convention, the functions* $\text{source}_{\mathscr{S}}$ *and* $\text{target}_{\mathscr{S}}$ *are defined in the obvious way. For example,* $\text{source}_{\mathscr{S}}(t_{02}) = s_0$ *and* $\text{target}_{\mathscr{S}}(t_{02}) = s_2$. *The function* $\text{prob}_{\mathscr{S}}$ *can be easily extracted from the above diagram. For example,* $\text{prob}_{\mathscr{S}}(t_{02}) = \frac{1}{2}$. *All states are labelled with the atomic proposition a and the states* s_1 *and* s_2 *are also labelled with the atomic proposition b. Hence, for example,* $\text{label}_{\mathscr{S}}(s_2) = \{a, b\}$.

Instead of $\langle S, T, AP, s_0, \text{source}, \text{target}, \text{prob}, \text{label}\rangle$ we usually write $\mathscr{S}$ and we denote, for example, its set of states by $S_{\mathscr{S}}$. We model the potential executions of the system under verification as execution paths of the PTS.

Definition 3 *An execution path of a PTS $\mathscr{S}$ is an infinite sequence of transitions $t_1 t_2 \ldots$ such that*

- *for all $i \geq 1$, $t_i \in T_{\mathscr{S}}$,*
- *$\text{source}_{\mathscr{S}}(t_1) = s_{0\mathscr{S}}$, and*
- *for all $i \geq 1$, $\text{target}_{\mathscr{S}}(t_i) = \text{source}_{\mathscr{S}}(t_{i+1})$.*

The set of all execution paths is denoted by $\text{Exec}_{\mathscr{S}}$.

Example 4 *Consider the PTS of Example 2. For this system, $t_{02}t_{22}{}^{\omega}$, $t_{01}t_{13}t_{33}{}^{\omega}$, and $t_{01}t_{10}t_{02}t_{22}{}^{\omega}$ are examples of execution paths.*

To define the progress measure, we use a measurable space of execution paths. We assume that the reader is familiar with the basics of measure theory as can be found in, for example, [3]. Recall that a measurable space consists of a set, a σ-algebra and a measure. In our case, the set is $\text{Exec}_{\mathscr{S}}$. The σ-algebra $\Sigma_{\mathscr{S}}$ is generated from the basic cylinder sets defined below. We denote the set of finite prefixes of execution paths in $\text{Exec}_{\mathscr{S}}$ by $\text{pref}(\text{Exec}_{\mathscr{S}})$.

Definition 5 *Let $e \in \text{pref}(\text{Exec}_{\mathscr{S}})$. Its basic cylinder set $B_{\mathscr{S}}^e$ is defined by*

$$B_{\mathscr{S}}^e = \{\, e' \in \text{Exec}_{\mathscr{S}} \mid e \text{ is a prefix of } e' \,\}.$$

The measure $\mu_{\mathscr{S}}$ is defined on a basic cylinder set $B_{\mathscr{S}}^{t_1 \ldots t_n}$ by

$$\mu_{\mathscr{S}}(B_{\mathscr{S}}^{t_1 \ldots t_n}) = \prod_{1 \leq i \leq n} \text{prob}_{\mathscr{S}}(t_i).$$

The measurable space $\langle \text{Exec}_{\mathscr{S}}, \Sigma_{\mathscr{S}}, \mu_{\mathscr{S}} \rangle$ is a sequence space as defined, for example, in [5, Chapter 2].

The verification effort of the probabilistic model checker is represented by its search of the PTS. The search is captured by the set of transitions that have been explored during the search.

Definition 6 *A search of a PTS $\mathscr{S}$ is a finite subset of $T_{\mathscr{S}}$.*

Example 7 *Consider the PTS of Example 2. The sets $\emptyset$, $\{t_{01}\}$, $\{t_{02}\}$, $\{t_{01}, t_{02}\}$ and $\{t_{01}, t_{02}, t_{10}, t_{13}, t_{22}, t_{33}\}$ are examples of searches.*

A PTS is said to extend a search if the transitions of the search are part of the PTS. We will use this notion in the definition of the progress measure.

Definition 8 *The PTS $\mathscr{S}'$ extends the search T of the PTS $\mathscr{S}$ if for all $t \in T$,*

- *$t \in T_{\mathscr{S}'}$,*
- *$s_{0\mathscr{S}} = s_{0\mathscr{S}'}$,*
- *$\text{source}_{\mathscr{S}'}(t) = \text{source}_{\mathscr{S}}(t)$,*
- *$\text{target}_{\mathscr{S}'}(t) = \text{target}_{\mathscr{S}}(t)$,*
- *$\text{prob}_{\mathscr{S}'}(t) = \text{prob}_{\mathscr{S}}(t)$,*
- *$\text{label}_{\mathscr{S}'}(\text{source}_{\mathscr{S}'}(t)) = \text{label}_{\mathscr{S}}(\text{source}_{\mathscr{S}}(t))$, and*

- $\text{label}_{\mathscr{S}'}(\text{target}_{\mathscr{S}'}(t)) = \text{label}_{\mathscr{S}}(\text{target}_{\mathscr{S}}(t))$.

Example 9 *Consider the PTS of Example 2 and the search* $\{t_{01}, t_{02}\}$. *The PTS*
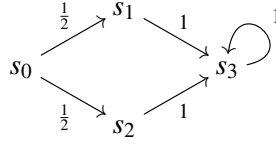
extends the search.

Since the PTSs we will consider in the remainder of this paper all extend a search T of a PTS $\mathscr{S}$, we write s_0 instead of $s_{0,\mathscr{S}}$ to avoid clutter. PTSs that extend a particular search give rise to the same set of execution paths if we restrict ourselves to those execution paths that only consist of transitions explored during the search.

Proposition 10 *If the PTS $\mathscr{S}'$ extends the search T of the PTS $\mathscr{S}$, then*

(a) $T^* \cap \text{pref}(\text{Exec}_{\mathscr{S}}) = T^* \cap \text{pref}(\text{Exec}_{\mathscr{S}'})$ *and*

(b) $T^{\omega} \cap \text{Exec}_{\mathscr{S}} = T^{\omega} \cap \text{Exec}_{\mathscr{S}'}$.

PTSs that extend a particular search also assign the same measure to basic cylinder sets of prefixes of execution paths only consisting of transitions explored during the search.

Proposition 11 *If the PTS $\mathscr{S}'$ extends the search T of the PTS $\mathscr{S}$, then $\mu_{\mathscr{S}}(B^e_{\mathscr{S}}) = \mu_{\mathscr{S}'}(B^e_{\mathscr{S}'})$ for all* $e \in T^* \cap \text{pref}(\text{Exec}_{\mathscr{S}})$.

The function $\text{label}_{\mathscr{S}}$ assigns to each state the set of atomic propositions that hold in the state. This function is extended to (prefixes of) execution paths as follows.

Definition 12 *The function* $\text{trace}_{\mathscr{S}} : \text{Exec}_{\mathscr{S}} \to (2^{AP_{\mathscr{S}}})^{\omega}$ *is defined by*

$$\text{trace}_{\mathscr{S}}(t_1 t_2 \ldots) = \text{label}_{\mathscr{S}}(\text{source}_{\mathscr{S}}(t_1))\text{label}_{\mathscr{S}}(\text{source}_{\mathscr{S}}(t_2))\ldots$$

The function $\text{trace}_{\mathscr{S}} : \text{pref}(\text{Exec}_{\mathscr{S}}) \to (2^{AP_{\mathscr{S}}})^*$ *is defined by*

$$\text{trace}_{\mathscr{S}}(t_1 \ldots t_n) = \text{label}_{\mathscr{S}}(\text{source}_{\mathscr{S}}(t_1))\ldots\text{label}_{\mathscr{S}}(\text{source}_{\mathscr{S}}(t_n))\text{label}_{\mathscr{S}}(\text{target}_{\mathscr{S}}(t_n))$$

Example 13 *Consider the PTS $\mathscr{S}$ of Example 2.*

$$\begin{aligned}
\text{trace}_{\mathscr{S}}(t_{02}t_{22}{}^{\omega}) &= \{a\}\{a,b\}^{\omega} \\
\text{trace}_{\mathscr{S}}(t_{01}t_{13}t_{33}{}^{\omega}) &= \{a\}\{a,b\}\{a\}^{\omega} \\
\text{trace}_{\mathscr{S}}(t_{01}t_{10}t_{02}t_{22}{}^{\omega}) &= \{a\}\{a,b\}\{a\}\{a,b\}^{\omega}
\end{aligned}$$

For the definition of linear-time property and the satisfaction relation $\models$ we refer the reader to, for example, [2, Section 3.2]. Based on these notions, we define when an execution path of a PTS satisfies a linear-time property.

Definition 14 *The satisfaction relation $\models_{\mathscr{S}}$ is defined by*

$$e \models_{\mathscr{S}} \phi \;\; \text{if trace}_{\mathscr{S}}(e) \models \phi.$$

For PTSs that extend a particular search, those execution paths that only consist of transitions explored by the search satisfy the same linear-time properties.

Proposition 15 *Let ϕ be a linear-time property. If the PTS $\mathscr{S}'$ extends the search T of the PTS $\mathscr{S}$, then $e \models_{\mathscr{S}} \phi$ iff $e \models_{\mathscr{S}'} \phi$ for all $e \in T^{\omega} \cap \mathrm{Exec}_{\mathscr{S}}$.*

Proof Since $\mathscr{S}'$ extends T of $\mathscr{S}$, $\mathrm{trace}_{\mathscr{S}}(e) = \mathrm{trace}_{\mathscr{S}'}(e)$ for all $e \in T^{\omega} \cap \mathrm{Exec}_{\mathscr{S}}$. $\square$

Next, we introduce the notion of a progress measure. Given a search of a PTS and a linear-time property, it captures the amount of progress the search of the probabilistic model checker has made towards verifying the linear-time property.

Definition 16 *Let the PTS $\mathscr{S}'$ extend the search T of PTS $\mathscr{S}$ and let ϕ be a linear-time property. The set $\mathscr{B}_{\mathscr{S}'}^{\phi}(T)$ is defined by*

$$\mathscr{B}_{\mathscr{S}'}^{\phi}(T) = \bigcup \{ B_{\mathscr{S}'}^{e} \mid e \in T^* \wedge \forall e' \in B_{\mathscr{S}'}^{e} : e' \models_{\mathscr{S}'} \phi \}.$$

The set $\mathscr{B}_{\mathscr{S}'}^{\phi}(T)$ is the union of those basic cylinder sets $B_{\mathscr{S}'}^{e}$ the execution paths of which satisfy the linear-time property ϕ. Hence, $B_{\mathscr{S}'}^{e}$ does not contain any execution paths violating ϕ. The set $\mathscr{B}_{\mathscr{S}'}^{\phi}(T)$ is measurable, as shown in [11, Proposition 1]. Hence, the measure $\mu_{\mathscr{S}'}$ assigns it a real number in the unit interval. This number represents the "size" of the basic cylinder sets that do not contain any violations of ϕ. This number captures the amount of progress of the search T verifying ϕ, *provided that* the PTS under consideration is $\mathscr{S}'$. However, we have no knowledge of the transitions other than the search. Therefore, we consider all extensions $\mathscr{S}'$ of T and consider the worst case in terms of progress.

Definition 17 *The progress of the search T of the PTS $\mathscr{S}$ of the linear-time property ϕ is defined by*

$$\mathrm{prog}_{\mathscr{S}}(T, \phi) = \inf \left\{ \mu_{\mathscr{S}'} \left(\mathscr{B}_{\mathscr{S}'}^{\phi}(T) \right) \mid \mathscr{S}' \text{ extends } T \text{ of } \mathscr{S} \right\}.$$

Example 18 *Consider the PTS $\mathscr{S}$ of Example 2 and the linear temporal logic formulae $\Box a$, $\Diamond a$, $\Diamond b$ and $\bigcirc b$. In the table below, we present the progress of these properties for a number of searches.*

search	$\Box a$	$\Diamond a$	$\Diamond b$	$\bigcirc b$
$\emptyset$	0	1	0	0
$\{t_{01}\}$	0	1	$\frac{1}{2}$	$\frac{1}{2}$
$\{t_{02}\}$	0	1	$\frac{1}{2}$	$\frac{1}{2}$
$\{t_{01}, t_{02}\}$	0	1	1	1
$\{t_{01}, t_{13}, t_{33}\}$	$\frac{1}{4}$	1	$\frac{1}{2}$	$\frac{1}{2}$
$\{t_{01}, t_{10}, t_{13}, t_{33}\}$	$\frac{1}{3}$	1	$\frac{1}{2}$	$\frac{1}{2}$

In [11, Theorem 1], Zhang and Van Breugel prove the following key property of their progress measure. They show that it is a lower bound for the probability that the linear-time property holds.

Theorem 19 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. Then*

$$\mathrm{prog}_{\mathscr{S}}(T, \phi) \leq \mu_{\mathscr{S}}(\{ e \in \mathrm{Exec}_{\mathscr{S}} \mid e \models_{\mathscr{S}} \phi \}).$$

The setting in this paper is slightly different from the one in [11]. In this paper we assume that PTSs do not have final states. This assumption can be made without loss of any generality: simply add a self loop with probability one to each final state.

3   Negation and Violations

In this section, we consider the relationship between making progress towards verifying a linear-time property and finding a violation of its negation. First, we formalize that a search has not found a violation of a linear-time property.

Definition 20 *The search T of the PTS $\mathscr{S}$ has not found a violation of the linear-time property ϕ if there exists a PTS $\mathscr{S}'$ which extends T of $\mathscr{S}$ such that $e \models_{\mathscr{S}'} \phi$ for all $e \in \mathrm{Exec}_{\mathscr{S}'}$.*

This definition is slightly stronger than the one given in [11, Definition 7]. All results of [11] remain valid for this stronger version. Next, we prove that if a search has made some progress towards verifying a linear-time property $\neg\phi$, then that search has also found a violation of ϕ.

Proposition 21 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. If $\mathrm{prog}_{\mathscr{S}}(T, \neg\phi) > 0$ then T has found a violation of ϕ.*

Proof By the definition of prog, $\mu_{\mathscr{S}'}(\mathscr{B}_{\mathscr{S}'}^{\neg\phi}(T)) > 0$ for each PTS $\mathscr{S}'$ which extends T of $\mathscr{S}$. Hence, $\mathscr{B}_{\mathscr{S}'}^{\neg\phi}(T) \neq \emptyset$. Therefore, there exists $e \in T^*$ such that $B_{\mathscr{S}'}^e \neq \emptyset$ and $\forall e' \in B_{\mathscr{S}'}^e : e' \models_{\mathscr{S}'} \neg\phi$. Hence, $e' \not\models_{\mathscr{S}'} \phi$ and $e' \in \mathrm{Exec}_{\mathscr{S}'}$. Therefore, T has found a violation of ϕ. $\square$

The reverse implication does not hold in general, as shown in the following example.

Example 22 *Consider the PTS*
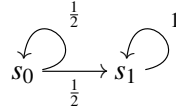
Assume that the state s_0 satisfies the atomic proposition a and the state s_1 does not. Consider the linear-time property $\square a$ and the search $\{t_{00}\}$. Note that $t_{00}{}^\omega \not\models \neg\square a$ and, hence, $\{t_{00}\}$ has found a violation of $\neg\square a$. Also note that $\mathrm{prog}_{\mathscr{S}}(\{t_{00}\}, \square a) = 0$.

We conjecture that the reverse implication does hold for safety properties (see, for example, [2, Definition 3.22] for a formal definition of safety property). However, so far we have only been able to prove it for invariants.

Proposition 23 *If the search T of the PTS $\mathscr{S}$ has found a violation of the invariant ϕ then $\mathrm{prog}_{\mathscr{S}}(T, \neg\phi) > 0$.*

Proof For every PTS $\mathscr{S}'$ that extends T, $e \not\models_{\mathscr{S}'} \square a$ for some $e \in \mathrm{Exec}_{\mathscr{S}'}$. Hence, $e = e_f t e_\ell$ for some $e_f \in T^* \cap \mathrm{pref}(\mathrm{Exec}_{\mathscr{S}'})$ and $t \in T$ such that $a \notin \mathrm{label}_{\mathscr{S}'}(\mathrm{source}_{\mathscr{S}'}(t))$. Therefore, for all $e' \in B_{\mathscr{S}'}^{e_f}$ we have that $e' \models_{\mathscr{S}'} \neg\square a$ and $B_{\mathscr{S}'}^{e_f} \neq \emptyset$. Hence, $\mu_{\mathscr{S}'}(B_{\mathscr{S}'}^{e_f}) > 0$ and, therefore, $\mathrm{prog}_{\mathscr{S}}(T, \neg\square a) > 0$. $\square$

4   A Positive Fragment of LTL

Next, we introduce a positive fragment of linear temporal logic (LTL). This fragment lacks negation. In Section 5 we will show how to compute the progress measure for this fragment.

Definition 24 *The logic LTL_+ is defined by*

$$\phi ::= \mathrm{true} \mid \mathrm{false} \mid a \mid \phi \wedge \phi \mid \phi \vee \phi \mid \bigcirc\phi \mid \phi_1 \, \mathscr{U} \, \phi_2 \mid \phi_1 \, \mathscr{R} \, \phi_2$$

where $a \in AP$.

The grammar defining LTL_+ is the same as the grammar defining the logic PNF introduced in [2, Definition 5.23], except that the grammar of LTL_+ does not contain $\neg a$. For each LTL formula, there exists an equivalent PNF formula (see, for example, [2, Section 5.1.5]). Such a result, of course, does not hold for LTL_+.

A property of LTL_+ that is key for our development is presented next.

Proposition 25 *For all LTL_+ formulae ϕ and $\sigma \in (2^{AP})^*$, $\sigma \emptyset^\omega \models \phi$ iff $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi$.*

Proof We prove two implications. Let ϕ be a LTL_+ formula and let $\sigma \in (2^{AP})^*$. Assume that $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi$. Since $\emptyset^\omega \in (2^{AP})^\omega$, we can immediately conclude that $\sigma\emptyset^\omega \models \phi$.

The other implication is proved by structural induction on ϕ. Let $\sigma \in (2^{AP})^*$. We distinguish the following cases.

- In case $\phi = \text{true}$, clearly $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi$ and, hence, the property is satisfied.
- In case $\phi = \text{false}$, obviously $\sigma\emptyset^\omega \models \phi$ is not satisfied and, therefore, the property holds.
- Let $\phi = a$. If $\sigma\emptyset^\omega \models \phi$, then $|\sigma| > 0$ and $a \in \sigma[0]$ and, hence, $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi$.
- Let $\phi = \phi_1 \wedge \phi_2$. Assume that $\sigma\emptyset^\omega \models \phi$. Then $\sigma\emptyset^\omega \models \phi_1$ and $\sigma\emptyset^\omega \models \phi_2$. By induction, $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi_1$ and $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi_2$. Hence, $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi$.
- The case $\phi = \phi_1 \vee \phi_2$ is similar to the previous case.
- For $\bigcirc\phi$ we distinguish the following two cases. Assume $|\sigma| = 0$. Suppose $\sigma\emptyset^\omega \models \bigcirc\phi$. Then $\emptyset^\omega[1\ldots] = \emptyset^\omega \models \phi$. By induction, $\forall \rho \in (2^{AP})^\omega : \rho \models \phi$. Hence, $\forall \rho \in (2^{AP})^\omega : \rho \models \bigcirc\phi$.
  Assume $|\sigma| \geq 1$. Suppose $\sigma\emptyset^\omega \models \bigcirc\phi$. Then $(\sigma\emptyset^\omega)[1\ldots] = \sigma[1\ldots]\emptyset^\omega \models \phi$. By induction, $\forall \rho \in (2^{AP})^\omega : \sigma[1\ldots]\rho \models \phi$. Since $\sigma[1\ldots]\rho = (\sigma\rho)[1\ldots]$, we have that $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \bigcirc\phi$.
- Next, let $\phi = \phi_1 \,\mathcal{U}\, \phi_2$. Assume that $\sigma\emptyset^\omega \models \phi$. Then there exists some $j \geq 0$ such that
  (a) $(\sigma\emptyset^\omega)[i\ldots] \models \phi_1$ for all $0 \leq i < j$ and
  (b) $(\sigma\emptyset^\omega)[j\ldots] \models \phi_2$.
  We distinguish two cases. Suppose $j < |\sigma|$. From (a) we can conclude that for all $0 \leq i < j$, $(\sigma\emptyset^\omega)[i\ldots] = \sigma[i\ldots]\emptyset^\omega \models \phi_1$. By induction, $\forall \rho \in (2^{AP})^\omega : \sigma[i\ldots]\rho \models \phi_1$. Since $\sigma[i\ldots]\rho = (\sigma\rho)[i\ldots]$, we have that $\forall \rho \in (2^{AP})^\omega : (\sigma\rho)[i\ldots] \models \phi_1$. From (b) we can deduce that $(\sigma\emptyset^\omega)[j\ldots] = \sigma[j\ldots]\emptyset^\omega \models \phi_2$. By induction, $\forall \rho \in (2^{AP})^\omega : \sigma[j\ldots]\rho \models \phi_2$. Since $\sigma[j\ldots]\rho = (\sigma\rho)[j\ldots]$, we have that $\forall \rho \in (2^{AP})^\omega : (\sigma\rho)[j\ldots] \models \phi_2$. Combining the above, we get $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi_1 \,\mathcal{U}\, \phi_2$. Suppose $j \geq |\sigma|$. For $0 \leq i < |\sigma|$, the argument for (a) is the same as above. For $|\sigma| \leq i < j$, (a) simply says that $\emptyset^\omega \models \phi_1$, which, by induction, implies that $\forall \rho \in (2^{AP})^\omega : \rho \models \phi_1$. Hence, $\forall \rho \in (2^{AP})^\omega : (\sigma\rho)[i\ldots] \models \phi_1$ for all $0 \leq i < j$. In this case, (b) means $\emptyset^\omega \models \phi_2$, which, by induction, implies that $\forall \rho \in (2^{AP})^\omega : \rho \models \phi_2$. Hence, $\forall \rho \in (2^{AP})^\omega : (\sigma\rho)[j\ldots] \models \phi_2$. Combining the above, we obtain that $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \phi_1 \,\mathcal{U}\, \phi_2$.
- Finally, we consider $\phi_1 \,\mathcal{R}\, \phi_2$. According to [2, page 256], $\phi_1 \,\mathcal{R}\, \phi_2 \equiv \neg(\neg\phi_1 \,\mathcal{U}\, \neg\phi_2)$ and $\neg(\phi_1 \,\mathcal{U}\, \phi_2) \equiv (\neg\phi_2) \,\mathcal{W}\, (\neg\phi_1 \wedge \neg\phi_2)$. According to [2, page 252], $\phi_1 \,\mathcal{W}\, \phi_2 \equiv (\phi_1 \,\mathcal{U}\, \phi_2) \vee \Box\phi_1$. Hence, we can derive that $\phi_1 \,\mathcal{R}\, \phi_2 \equiv (\phi_2 \,\mathcal{U}\, (\phi_1 \wedge \phi_2)) \vee \Box\phi_2$. Therefore, proving that the property is satisfied by $\Box\phi$, combined with the proofs for $\wedge$, $\vee$ and $\mathcal{U}$ above, suffices as proof for $\phi_1 \,\mathcal{R}\, \phi_2$. Thus, we consider $\Box\phi$. Suppose that $\sigma\emptyset^\omega \models \Box\phi$. Then $(\sigma\emptyset^\omega)[j\ldots] \models \phi$ for all $j \geq 0$. We distinguish two cases. For all $0 \leq j < |\sigma|$, we have that $(\sigma\emptyset^\omega)[j\ldots] = \sigma[j\ldots]\emptyset^\omega \models \phi$. By induction, $\forall \rho \in (2^{AP})^\omega : \sigma[j\ldots]\rho \models \phi$ and, hence, $\forall \rho \in (2^{AP})^\omega : (\sigma\rho)[j\ldots] \models \phi$.
  For all $j \geq |\sigma|$, we have that $(\sigma\emptyset^\omega)[j\ldots] = \emptyset^\omega \models \phi$. By induction, $\forall \rho \in (2^{AP})^\omega : \rho \models \phi$ and, therefore, $\forall \rho \in (2^{AP})^\omega : (\sigma\rho)[j\ldots] \models \phi$. Combining the above, we get $\forall \rho \in (2^{AP})^\omega : \sigma\rho \models \Box\phi$.

□

The above result does not hold for all LTL formulae, as shown in the following example.

Example 26 *Consider the LTL formula $\neg a$. Note that this formula is not equivalent to any LTL_+ formula. Let $\sigma = \varepsilon$. Obviously, $\emptyset^\omega \models \neg a$, but it is not the case that $\forall \rho \in (2^{AP})^\omega : \rho \models \neg a$ (just take a $\rho \in (2^{AP})^\omega$ with $a \in \rho[0]$).*

5   An Algorithm to Compute Progress

To obtain an algorithm to compute the progress for the positive fragment of LTL, we present an alternative characterization of the progress measure. This alternative characterization is cast in terms of a PTS built from the search as follows. We start from the transitions of the search and their source and target states. We add a sink state, which has a transition to itself with probability one and which does not satisfy any atomic proposition. For each state which has not been fully explored yet, that is, the sum of the probabilities of its outgoing transitions is less than one, we add a transition to the sink state with the remaining probability. This PTS can be viewed as the minimal extension of the search (we will formalize this in Proposition 34). The PTS is defined as follows.

Definition 27 *Let T be a search of the PTS $\mathscr{S}$. The set $S^T_{\mathscr{S}}$ is defined by*

$$S^T_{\mathscr{S}} = \{\, \text{source}_{\mathscr{S}}(t) \mid t \in T \,\} \cup \{\, \text{target}_{\mathscr{S}}(t) \mid t \in T \,\} \cup \{s_0\}.$$

For each $s \in S^T_{\mathscr{S}}$,

$$\text{out}_{\mathscr{S}}(s) = \sum \{\, \text{prob}_{\mathscr{S}}(t) \mid t \in T \wedge \text{source}_{\mathscr{S}}(t) = s \,\}.$$

The PTS $\mathscr{S}_T$ is defined by

- $S_{\mathscr{S}_T} = S^T_{\mathscr{S}} \cup \{s_\perp\}$,
- $T_{\mathscr{S}_T} = T \cup \{\, t_s \mid s \in S^T_{\mathscr{S}} \wedge \text{out}_{\mathscr{S}}(s) < 1 \,\} \cup \{t_\perp\}$,
- $\text{source}_{\mathscr{S}_T}(t) = \begin{cases} \text{source}_{\mathscr{S}}(t) & \text{if } t \in T \\ s & \text{if } t = t_s \\ s_\perp & \text{if } t = t_\perp \end{cases}$
- $\text{target}_{\mathscr{S}_T}(t) = \begin{cases} \text{target}_{\mathscr{S}}(t) & \text{if } t \in T \\ s_\perp & \text{if } t = t_\perp \text{ or } t = t_s \end{cases}$
- $\text{prob}_{\mathscr{S}_T}(t) = \begin{cases} \text{prob}_{\mathscr{S}}(t) & \text{if } t \in T \\ 1 - \text{out}_{\mathscr{S}}(s) & \text{if } t = t_s \\ 1 & \text{if } t = t_\perp \end{cases}$
- $\text{label}_{\mathscr{S}_T}(s) = \begin{cases} \emptyset & \text{if } s = s_\perp \\ \text{label}_{\mathscr{S}}(s) & \text{otherwise} \end{cases}$

The above definition is very similar to [11, Definition 10]. The main difference is that we do not have final states.

Proposition 28 *Let T be a search of the PTS $\mathscr{S}$. Then the PTS $\mathscr{S}_T$ extends T.*

Proof Follows immediately from the definition of $\mathscr{S}_T$. $\square$

Next, we will show that the PTS $\mathscr{S}_T$ is the minimal extension of the search T of the PTS $\mathscr{S}$. More precisely, we will prove that for any other extension $\mathscr{S}'$ of T we have that $\mu_{\mathscr{S}_T}(\mathscr{B}^\phi_{\mathscr{S}_T}) \leq \mu_{\mathscr{S}'}(\mathscr{B}^\phi_{\mathscr{S}'})$. To prove this result, we introduce two new notions and some of their properties.

Definition 29 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. The set $E^\phi_{\mathscr{S}}(T)$ is defined by*

$$E^\phi_{\mathscr{S}}(T) = \{\, e \in T^* \cap \mathrm{pref}(\mathrm{Exec}_{\mathscr{S}}) \mid \forall e' \in B^e_{\mathscr{S}} : e' \models_{\mathscr{S}} \phi \,\}.$$

The set $E^\phi_{\mathscr{S}_T}(T)$ is minimal among the $E^\phi_{\mathscr{S}'}(T)$ where $\mathscr{S}'$ extends T.

Proposition 30 *Let the PTS $\mathscr{S}'$ extend the search T of the PTS $\mathscr{S}$. For any LTL$_+$ formula ϕ, $E^\phi_{\mathscr{S}_T}(T) \subseteq E^\phi_{\mathscr{S}'}(T)$.*

Next, we restrict our attention to those elements of $E^\phi_{\mathscr{S}}(T)$ which are minimal with respect to the prefix order.

Definition 31 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. The set $ME^\phi_{\mathscr{S}}(T)$ is defined by*

$$ME^\phi_{\mathscr{S}}(T) = \{\, e \in E^\phi_{\mathscr{S}}(T) \mid |e| > 0 \Rightarrow \exists e' \in B^{e[|e|-1]}_{\mathscr{S}} : e' \not\models_{\mathscr{S}} \phi \,\}.$$

Note that $e \in ME^\phi_{\mathscr{S}}(T)$ if and only if it belongs to $E^\phi_{\mathscr{S}}(T)$ and none of its prefixes belong to $E^\phi_{\mathscr{S}}(T)$.

Proposition 32 *Let the PTSs $\mathscr{S}'$ and $\mathscr{S}''$ extend the search T of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. Then*

$$\bigcup_{\bar{e} \in ME^\phi_{\mathscr{S}''}(T)} B^{\bar{e}}_{\mathscr{S}'} = \bigcup_{e \in E^\phi_{\mathscr{S}''}(T)} B^e_{\mathscr{S}'}.$$

Proof Since $ME^\phi_{\mathscr{S}''}(T) \subseteq E^\phi_{\mathscr{S}''}(T)$, we can conclude that the set on the left hand side is a subset of the set on the right hand side. Next, we prove the other inclusion. We show that for each $e \in E^\phi_{\mathscr{S}''}(T)$ there exists $\bar{e} \in ME^\phi_{\mathscr{S}''}(T)$ such that $B^e_{\mathscr{S}'} \subseteq B^{\bar{e}}_{\mathscr{S}'}$ by induction on the length of e. In the base case, $|e| = 0$, then $e \in E^\phi_{\mathscr{S}''}(T)$ implies $e \in ME^\phi_{\mathscr{S}''}(T)$ and, hence, we take $\bar{e}$ to be e. Let $|e| > 0$. We distinguish two cases. If $\exists e' \in B^{e[|e|-1]}_{\mathscr{S}'} : e' \not\models_{\mathscr{S}} \phi$ then we also take $\bar{e}$ to be e. Otherwise, $e[|e|-1] \in E^\phi_{\mathscr{S}''}(T)$. Obviously, $B^e_{\mathscr{S}'} \subseteq B^{e[|e|-1]}_{\mathscr{S}'}$ and, by induction, there exists a $\bar{e} \in ME^\phi_{\mathscr{S}''}(T)$ such that $B^{e[|e|-1]}_{\mathscr{S}'} \subseteq B^{\bar{e}}_{\mathscr{S}'}$. $\qquad\square$

Proposition 33 *Let the PTSs $\mathscr{S}'$ and $\mathscr{S}''$ extend the search T of the PTS $\mathscr{S}$, and let ϕ be a linear-time property. If $E^\phi_{\mathscr{S}'}(T) \subseteq E^\phi_{\mathscr{S}''}(T)$ then*

$$\mu_{\mathscr{S}''}\left(\bigcup\{\, B^e_{\mathscr{S}''} \mid e \in ME^\phi_{\mathscr{S}'}(T) \,\}\right) = \sum_{e \in ME^\phi_{\mathscr{S}'}(T)} \mu_{\mathscr{S}''}(B^e_{\mathscr{S}''}). \qquad (1)$$

Proof We have that

$$
\begin{aligned}
ME^\phi_{\mathscr{S}'}(T) \;&\subseteq\; E^\phi_{\mathscr{S}'}(T) \quad \text{[by definition]} \\
&\subseteq\; E^\phi_{\mathscr{S}''}(T) \quad \text{[by assumption]} \\
&\subseteq\; \mathrm{pref}(\mathrm{Exec}_{\mathscr{S}''}) \quad \text{[by definition]}
\end{aligned}
$$

Hence, for all $e \in ME^\phi_{\mathscr{S}'}(T)$, we have that $B^e_{\mathscr{S}''} \in \Sigma_{\mathscr{S}''}$. Since the set T is finite, the set T^* is countable and, hence, the set $ME^\phi_{\mathscr{S}'}(T)$ is countable as well. Since a σ-algebra is closed under countable unions, $\bigcup\{\, B^e_{\mathscr{S}''} \mid e \in ME^\phi_{\mathscr{S}'}(T) \,\} \in \Sigma_{\mathscr{S}''}$. Hence, the measure $\mu_{\mathscr{S}''}$ is defined on this set.

To conclude (1), it suffices to prove that for all $e_1, e_2 \in ME^\phi_{\mathscr{S}'}(T)$ such that $e_1 \neq e_2$, e_1 is not a prefix of e_2, since this implies that $B^{e_1}_{\mathscr{S}'}$ and $B^{e_2}_{\mathscr{S}'}$ are disjoint. Towards a contradiction, assume that e_1 is a prefix of e_2. Since $\forall e'_1 \in B^{e_1}_{\mathscr{S}'} : e'_1 \models_{\mathscr{S}} \phi$ and e_1 is a prefix of e_2 and $e_1 \neq e_2$, it cannot be the case that $\exists e'_2 \in B^{e_2[|e_2|-1]}_{\mathscr{S}'} : e'_2 \not\models_{\mathscr{S}} \phi$. This contradicts the assumption that $e_2 \in ME^\phi_{\mathscr{S}'}(T)$. $\qquad\square$

Now, we are ready to prove that the PTS $\mathscr{S}_T$ is the minimal extension of the search T of the PTS $\mathscr{S}$.

Proposition 34 *Let the PTS $\mathscr{S}'$ extend the search T of the PTS $\mathscr{S}$ and let ϕ be a LTL$_+$ formula. Then*

$$\mu_{\mathscr{S}_T}(\mathscr{B}^{\phi}_{\mathscr{S}_T}(T)) \leq \mu_{\mathscr{S}'}(\mathscr{B}^{\phi}_{\mathscr{S}'}(T)).$$

Proof

$$
\begin{aligned}
&\mu_{\mathscr{S}_T}(\mathscr{B}^{\phi}_{\mathscr{S}_T}(T)) \\
&= \mu_{\mathscr{S}_T}(\bigcup\{B^e_{\mathscr{S}_T} \mid e \in E^{\phi}_{\mathscr{S}_T}(T)\}) \\
&= \mu_{\mathscr{S}_T}(\bigcup\{B^e_{\mathscr{S}_T} \mid e \in ME^{\phi}_{\mathscr{S}_T}(T)\}) \quad \text{[Proposition 32]} \\
&= \sum_{e \in ME^{\phi}_{\mathscr{S}_T}(T)} \mu_{\mathscr{S}_T}(B^e_{\mathscr{S}_T}) \quad \text{[Proposition 33]} \\
&= \sum_{e \in ME^{\phi}_{\mathscr{S}_T}(T)} \mu_{\mathscr{S}'}(B^e_{\mathscr{S}'}) \quad \text{[Proposition 11]} \\
&= \mu_{\mathscr{S}'}(\bigcup\{B^e_{\mathscr{S}'} \mid e \in ME^{\phi}_{\mathscr{S}_T}(T)\}) \quad \text{[Proposition 30 and 33]} \\
&= \mu_{\mathscr{S}'}(\bigcup\{B^e_{\mathscr{S}'} \mid e \in E^{\phi}_{\mathscr{S}_T}(T)\}) \quad \text{[Proposition 32]} \\
&\leq \mu_{\mathscr{S}'}(\bigcup\{B^e_{\mathscr{S}'} \mid e \in E^{\phi}_{\mathscr{S}'}(T)\}) \quad \text{[Proposition 30]} \\
&= \mu_{\mathscr{S}'}(\mathscr{B}^{\phi}_{\mathscr{S}'}(T))
\end{aligned}
$$

$\square$

The above proposition gives us an alternative characterization of the progress measure.

Theorem 35 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a LTL$_+$ formula. Then*

$$\mathrm{prog}_{\mathscr{S}}(T,\phi) = \mu_{\mathscr{S}_T}(\mathscr{B}^{\phi}_{\mathscr{S}_T}(T)).$$

Proof This is a direct consequence of the definition of the progress measure and Proposition 34. $\square$

Hence, in order to compute $\mathrm{prog}_{\mathscr{S}}(T,\phi)$, it suffices to compute the measure of $\mathscr{B}^{\phi}_{\mathscr{S}_T}(T)$. Next, we will show that the latter is equal to the measure of the set of execution paths of $\mathscr{S}_T$ that satisfy ϕ. The proof consists of two parts. First, we prove the following inclusion.
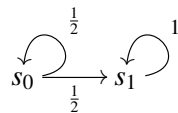
Proposition 36 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. Then*

$$\mathscr{B}^{\phi}_{\mathscr{S}_T}(T) \subseteq \{e \in \mathrm{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi\}.$$
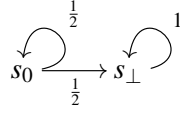
Proof Let $e \in \mathscr{B}^{\phi}_{\mathscr{S}_T}(T)$. Then $e \in B^{e'}_{\mathscr{S}_T}$ for some $e' \in T^*$ such that $\forall e'' \in B^{e'}_{\mathscr{S}_T} : e'' \models_{\mathscr{S}_T} \phi$. Hence, $e \models_{\mathscr{S}_T} \phi$. $\square$

The opposite inclusion does not hold in general, as shown in the following example.

Example 37 *Consider the PTS $\mathscr{S}$*

Consider the search $\{t_{00}\}$. Then the PTS $\mathscr{S}_T$ can be depicted by

$$s_0 \xrightarrow[\frac{1}{2}]{\frac{1}{2}} s_\perp \quad 1$$

Assume that the state s_0 satisfies the atomic proposition a. Hence, $t_{00}{}^\omega \models_{\mathscr{S}_T} \Box a$. By construction, the state $s_\perp$ does not satisfy a. Therefore, $t_{00}{}^\omega \notin \mathscr{B}_{\mathscr{S}_T}^{\Box a}$.

However, we will show that the set $\{\, e \in \mathrm{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\} \setminus \mathscr{B}_{\mathscr{S}_T}^{\phi}(T)$ has measure zero. In the proof, we will use the following proposition.

Proposition 38 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a linear-time property. Assume that T has not found a violation of ϕ. Then for all $e \in T^\omega \cap \mathrm{Exec}_{\mathscr{S}_T}$, $e \models_{\mathscr{S}_T} \phi$.*

Proof Let $e \in T^\omega \cap \mathrm{Exec}_{\mathscr{S}_T}$. Since T has not found a violation of ϕ, by definition there exists a PTS $\mathscr{S}'$ that extends T of $\mathscr{S}$ such that $e' \models_{\mathscr{S}'} \phi$ for all $e' \in \mathrm{Exec}_{\mathscr{S}'}$. Then $e \in \mathrm{Exec}_{\mathscr{S}'} \cap T^\omega$ by Proposition 10(b), because $\mathscr{S}'$ and $\mathscr{S}_T$ both extend T. Hence, $e \models_{\mathscr{S}'} \phi$. Therefore, from Proposition 15 we can conclude that $e \models_{\mathscr{S}_T} \phi$. $\qquad\square$

Proposition 39 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a LTL_+ formula. If T has not found a violation of ϕ then*

$$\mu_{\mathscr{S}_T}(\{\, e \in \mathrm{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\} \setminus \mathscr{B}_{\mathscr{S}_T}^{\phi}(T)) = 0.$$

Proof To avoid clutter, we denote the set $\{\, e \in \mathrm{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\} \setminus \mathscr{B}_{\mathscr{S}_T}^{\phi}(T)$ by Z.

First, we show that $Z \subseteq T^\omega$. Assume that $e \in Z$. Towards a contradiction, suppose that $e \notin T^\omega$. From the construction of $\mathscr{S}_T$ we can deduce that $e = e' t_s t_\perp{}^\omega$ for some $e' \in T^*$. Let $\mathrm{trace}_{\mathscr{S}_T}(e') = \sigma$. Then $\mathrm{trace}_{\mathscr{S}_T}(e) = \sigma \emptyset^\omega$. Since $e \in Z$, we have that $e \models_{\mathscr{S}_T} \phi$ and, hence, $\sigma \emptyset^\omega \models \phi$. By Proposition 25, $\forall \rho \in (2^{AP})^\omega : \sigma \rho \models \phi$. Hence, $\forall e'' \in B_{\mathscr{S}_T}^{e'} : e'' \models_{\mathscr{S}_T} \phi$. Since $e \in B_{\mathscr{S}_T}^{e'}$, we have that $e \in \mathscr{B}_{\mathscr{S}_T}^{\phi}(T)$, which contradicts our assumption that $e \in Z$.

Next, we show that each state in $\{\, \mathrm{target}_{\mathscr{S}_T}(e) \mid e \in \mathrm{pref}(Z) \,\}$ is transient. Roughly speaking, a state s is transient if the probability of reaching s in one or more transitions when starting in s is strictly less than one (see, for example, [1, Section 7.3] for a formal definition). It suffices to show that each state in $\{\, \mathrm{target}_{\mathscr{S}_T}(e) \mid e \in \mathrm{pref}(Z) \,\}$ can reach the state $s_\perp$, since in that case the probability of reaching $s_\perp$ and, hence, not returning to the state itself, is greater than zero.

Since T has not found a violation of ϕ, we can conclude from Proposition 38 that $e \models_{\mathscr{S}_T} \phi$ for all $e \in T^\omega$. Hence, from the construction of $\mathscr{S}_T$ we can deduce that if $e \not\models_{\mathscr{S}_T} \phi$ then $e \notin T^\omega$ and, hence, e reaches $s_\perp$. Let $e \in \mathrm{pref}(Z)$. Hence, there exists $e' \in B_{\mathscr{S}_T}^{e}$ such that $e' \not\models_{\mathscr{S}_T} \phi$. Therefore, e' reaches $s_\perp$ and, hence, $\mathrm{target}_{\mathscr{S}_T}(e)$ can reach $s_\perp$.

Since $Z \subseteq T^\omega$, the set $\{\, \mathrm{target}_{\mathscr{S}_T}(e) \mid e \in \mathrm{pref}(Z) \,\}$ is finite. According to [1, page 223], the probability of remaining in a finite set of transient states is zero. As a consequence, the probability of remaining in the set $\{\, \mathrm{target}_{\mathscr{S}'}(e) \mid e \in \mathrm{pref}(Z) \,\}$ is zero. Hence, we can conclude that $\mu_{\mathscr{S}_T}(Z) = 0$. $\qquad\square$

From the above, we can derive the following result.

Theorem 40 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a LTL_+ formula. If T has not found a violation of ϕ then*

$$\mu_{\mathscr{S}_T}(\mathscr{B}_{\mathscr{S}_T}^{\phi}(T)) = \mu_{\mathscr{S}_T}(\{\, e \in \mathrm{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\}).$$

Proof

$$\mu_{\mathscr{S}_T}(\mathscr{B}^\phi_{\mathscr{S}_T}(T))$$
$$\le \quad \mu_{\mathscr{S}_T}(\{\, e \in \text{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\}) \quad \text{[Proposition 36 and } \mu_{\mathscr{S}_T} \text{ is monotone]}$$
$$= \quad \mu_{\mathscr{S}_T}(\mathscr{B}^\phi_{\mathscr{S}_T}(T)) + \mu_{\mathscr{S}_T}(\{\, e \in \text{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\} \setminus \mathscr{B}^\phi_{\mathscr{S}_T}(T)) \quad \text{[Proposition 36 and } \mu_{\mathscr{S}_T} \text{ is additive]}$$
$$= \quad \mu_{\mathscr{S}_T}(\mathscr{B}^\phi_{\mathscr{S}_T}(T)) \quad \text{[Proposition 39]}$$

$\square$

Combining Theorem 35 and 40, we obtain the following characterization of the progress measure.

Corollary 41 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a LTL$_+$ formula. If T has not found a violation of ϕ then*

$$\text{prog}_{\mathscr{S}}(T,\phi) = \mu_{\mathscr{S}_T}(\{\, e \in \text{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\}).$$

Proof Immediate consequence of Theorem 35 and 40. $\square$

How to compute $\mu_{\mathscr{S}_T}(\{\, e \in \text{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \,\})$ can be found, for example, in [4, Section 3.1]. Computing this measure is exponential in the size of ϕ and polynomial in the size of T.

6 An Algorithm to Efficiently Compute a Lower Bound of Progress

The algorithm developed in the previous section to compute $\text{prog}_{\mathscr{S}}(T,\phi)$ is exponential in the size of ϕ. In this section, we trade precision for efficiency. We present an algorithm that does not compute $\text{prog}_{\mathscr{S}}(T,\phi)$, but only provides a lower bound in polynomial time. This lower bound is tight for invariants. However, we also show an example in which the lower bound does not provide us any information.

Next, we show that subsets of $\text{Exec}_{\mathscr{S}}$ can be characterized as countable intersections of countable unions of basic cylinder sets. For $A \subseteq \text{Exec}_{\mathscr{S}}$ and $n \in \mathbb{N}$, we use $A[n]$ to denote the set $\{\, e[n] \mid e \in A \,\}$, where $e[n]$ denotes the execution path e truncated at length n. We prove the characterization by showing two inclusions. The first inclusion holds for arbitrary subsets of $\text{Exec}_{\mathscr{S}}$.

Proposition 42 *For PTS $\mathscr{S}$, let $A \subseteq \text{Exec}_{\mathscr{S}}$. Then*

$$A \subseteq \bigcap_{n \in \mathbb{N}} \bigcup_{e \in A[n]} B^e_{\mathscr{S}}.$$

Proof Let $e' \in A$. It suffices to show that

$$e' \in \bigcup_{e \in A[n]} B^e_{\mathscr{S}} \tag{2}$$

for all $n \in \mathbb{N}$. Let $n \in \mathbb{N}$. To prove (2), it suffices to show that $e' \in B^e_{\mathscr{S}}$ for some $e \in A[n]$. Since $e' \in A$, we have that $e'[n] \in A[n]$. Because $e'[n]$ is a prefix of e' and $e' \in \text{Exec}_{\mathscr{S}}$, we have that $e' \in B^{e'[n]}_{\mathscr{S}}$, which concludes our proof. $\square$

The reverse inclusion does not hold in general. In some of the proofs below we use some metric topology. Those readers unfamiliar with metric topology are referred to, for example, [8]. To prove the reverse inclusion, we use that the set is closed.

Proposition 43 *For PTS $\mathscr{S}$, let $A \subseteq \text{Exec}_{\mathscr{S}}$. If A is closed then*

$$\bigcap_{n \in \mathbb{N}} \bigcup_{e \in A[n]} B^e_{\mathscr{S}} \subseteq A.$$

Proof Let $e' \in \bigcap_{n \in \mathbb{N}} \bigcup_{e \in A[n]} B^e_{\mathscr{S}}$. Then $e' \in \bigcup_{e \in A[n]} B^e_{\mathscr{S}}$ for all $n \in \mathbb{N}$. Hence, for each $n \in \mathbb{N}$ there exists a $e_n \in A[n]$ such that $e' \in B^{e_n}_{\mathscr{S}}$. Thus, for each $n \in \mathbb{N}$ there exists a $e'_n \in A$ such that $e' \in B^{e'_n[n]}_{\mathscr{S}}$ and, hence, $e'_n[n]$ is a prefix of e'.

We distinguish two cases. Assume that for some $n \in \mathbb{N}$, $e'_n[n] = e'_n$. Then e'_n is a prefix of e'. Since also $e', e'_n \in \text{Exec}_{\mathscr{S}}$, we can conclude that $e' = e'_n$. Since $e'_n \in A$ we have that $e' \in A$.

Otherwise, $e'_n[n] \neq e'_n$ for all $n \in \mathbb{N}$. Since also $e'_n[n]$ is a prefix of e', we can conclude that $e'_n[n] = e'[n]$. Let the distance function $d : (\text{pref}(\text{Exec}_{\mathscr{S}}) \cup \text{Exec}_{\mathscr{S}}) \times (\text{pref}(\text{Exec}_{\mathscr{S}}) \cup \text{Exec}_{\mathscr{S}}) \to [0,1]$ be defined by $d(e_1, e_2) = \inf\{ 2^{-n} \mid e_1[n] = e_2[n] \}$. Then, $d(e'_n, e') \leq 2^{-n}$, that is, the sequence $(e'_n)_n$ converges to e'. Because all the elements of the sequence $(e'_n)_n$ are in A and A is closed, we can conclude that the limit e' is in A as well (see, for example, [8, Proposition 3.7.15 and Lemma 7.2.2]). $\qquad\square$

PTSs that extend a particular search assign the same measure to closed sets of execution paths consisting only of explored transitions.

Proposition 44 *Let the PTS $\mathscr{S}'$ extend the search T of the PTS $\mathscr{S}$ and let $A \subseteq T^\omega \cap \text{Exec}_{\mathscr{S}}$. If A is closed then $\mu_{\mathscr{S}}(A) = \mu_{\mathscr{S}'}(A)$.*

Proof Obviously, for all $e \in T^*$ and $t \in T$, we have $B^e_{\mathscr{S}} \supseteq B^{et}_{\mathscr{S}}$. As a consequence, $\bigcup_{e \in A[n]} B^e_{\mathscr{S}} \supseteq \bigcup_{e \in A[n+1]} B^e_{\mathscr{S}}$ for all $n \in \mathbb{N}$. Furthermore, $\mu_{\mathscr{S}}(\bigcup_{e \in A[0]} B^e_{\mathscr{S}}) = \mu_{\mathscr{S}}(B^\varepsilon_{\mathscr{S}}) = 1$ and, hence, $\mu_{\mathscr{S}}(\bigcup_{e \in A[0]} B^e_{\mathscr{S}})$ is finite. Since a measure is continuous (see, for example, [3, Theorem 2.1]), we can conclude from the above that

$$\mu_{\mathscr{S}}\left(\bigcap_{n \in \mathbb{N}} \bigcup_{e \in A[n]} B^e_{\mathscr{S}} \right) = \lim_{n \in \mathbb{N}} \mu_{\mathscr{S}}\left(\bigcup_{e \in A[n]} B^e_{\mathscr{S}} \right). \tag{3}$$

Therefore,

$$
\begin{aligned}
\mu_{\mathscr{S}}(A) &= \mu_{\mathscr{S}}\left(\bigcap_{n \in \mathbb{N}} \bigcup_{e \in A[n]} B^e_{\mathscr{S}} \right) && \text{[Proposition 42 and 43]} \\
&= \lim_{n \in \mathbb{N}} \mu_{\mathscr{S}}\left(\bigcup_{e \in A[n]} B^e_{\mathscr{S}} \right) && \text{[(3)]} \\
&= \lim_{n \in \mathbb{N}} \sum_{e \in A[n]} \mu_{\mathscr{S}}(B^e_{\mathscr{S}}) && \text{[a measure is countably additive]} \\
&= \lim_{n \in \mathbb{N}} \sum_{t_1 \dots t_n \in A[n]} \prod_{1 \leq i \leq n} \text{prob}_{\mathscr{S}}(t_i) \\
&= \lim_{n \in \mathbb{N}} \sum_{t_1 \dots t_n \in A[n]} \prod_{1 \leq i \leq n} \text{prob}_{\mathscr{S}'}(t_i) && \text{[$\mathscr{S}'$ extends T of $\mathscr{S}$]} \\
&= \mu_{\mathscr{S}'}(A) && \text{[by symmetric argument].}
\end{aligned}
$$

$\qquad\square$

Hence, the PTSs $\mathscr{S}$ and $\mathscr{S}_T$ assign the same measure to the closed set of those execution paths consisting only of explored transitions.

Corollary 45 *Let T be a search of the PTS $\mathscr{S}$. Then $\mu_{\mathscr{S}}(T^{\omega} \cap \operatorname{Exec}_{\mathscr{S}}) = \mu_{\mathscr{S}_T}(T^{\omega} \cap \operatorname{Exec}_{\mathscr{S}_T})$.*

Proof Since the sets $\operatorname{Exec}_{\mathscr{S}}$ and T^{ω} are closed, their intersection is also closed (see, for example, [8, Proposition 3.7.5]) and, hence, the result follows immediately from Proposition 44 and 10(b). □

Now we can show that the measure of the set of execution paths consisting only of explored transitions is a lower bound for the progress measure.

Theorem 46 *Let T be a search of the PTS $\mathscr{S}$ and let ϕ be a LTL_+ formula. If T has not found a violation of ϕ then*

$$\mu_{\mathscr{S}_T}(T^{\omega} \cap \operatorname{Exec}_{\mathscr{S}_T}) \leq \operatorname{prog}_{\mathscr{S}}(T, \phi).$$

Proof

$$
\begin{aligned}
& \mu_{\mathscr{S}_T}(T^{\omega} \cap \operatorname{Exec}_{\mathscr{S}_T}) \\
\leq \quad & \mu_{\mathscr{S}_T}(\{ e \in \operatorname{Exec}_{\mathscr{S}_T} \mid e \models_{\mathscr{S}_T} \phi \}) \quad \text{[Proposition 38]} \\
= \quad & \operatorname{prog}_{\mathscr{S}}(T, \phi) \quad \text{[Corollary 41]}
\end{aligned}
$$

□

From the construction of $\mathscr{S}_T$ we can conclude that $\mu_{\mathscr{S}_T}(T^{\omega} \cap \operatorname{Exec}_{\mathscr{S}_T})$ is the same as $\mu_{\mathscr{S}_T}(\{ e \in \operatorname{Exec}_{\mathscr{S}_T} \mid e \text{ does not reach } s_{\perp} \})$, which is the same as $1 - \mu_{\mathscr{S}_T}(\{ e \in \operatorname{Exec}_{\mathscr{S}_T} \mid e \text{ reaches } s_{\perp} \})$. The latter can be computed in polynomial time using, for example, Gaussian elimination (see, for example, [2, Section 10.1.1]). This algorithm has been implemented and incorporated into an extension of the model checker JPF [10]. While JPF is model checking sequential Java code which contains probabilistic choices, our extension also keeps track of the underlying PTS. The amount of memory needed to store this PTS is in general only a small fraction of the total amount of memory needed. Once our extension of JPF runs almost out of memory, it can usually free enough memory so that the progress can be computed from the stored PTS.

As was shown in [11, Theorem 4], the above bound is tight for invariants.

Proposition 47 *If the search T of the PTS $\mathscr{S}$ has not found a violation of invariant ϕ then*

$$\mu_{\mathscr{S}_T}(T^{\omega} \cap \operatorname{Exec}_{\mathscr{S}_T}) = \operatorname{prog}_{\mathscr{S}}(T, \phi).$$

In the example below, we present a search of a PTS for a LTL_+ formula of which the progress is one whereas the bound is zero. In this case, the bound does not provide us any information.

Example 48 *Consider the PTS*
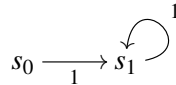
Assume that the state s_1 satisfies the atomic proposition a. Consider the linear-time property $\bigcirc a$ and the search $\{t_{01}\}$. In this case, we have that $\operatorname{prog}_{\mathscr{S}}(\{t_{01}\}, \bigcirc a) = 1$ but $\mu_{\mathscr{S}_{\{t_{01}\}}}(\{t_{01}\}^{\omega} \cap \operatorname{Exec}_{\mathscr{S}_{\{t_{01}\}}}) = \mu_{\mathscr{S}_{\{t_{01}\}}}(\emptyset) = 0$.

7 Conclusion

Our work is based on the paper by Zhang and Van Breugel [11]. The work by Pavese, Braberman and Uchitel [6] is also related. They aim to measure the probability that a run of the system reaches a state that has not been visited by the model checker. Also the work by Della Penna et al. [7] seems related. They show how, given a Markov chain and an integer i, the probability of reaching a particular state s within i transitions can be computed.

As we have seen, there seems to be a trade off between efficiency and accuracy when it comes to computing progress. Our algorithm to compute $\text{prog}_{\mathscr{S}}(T, \phi)$ is exponential in the size of the LTL$_+$ formula ϕ and polynomial in the size of the search T. We even conjecture (and leave it to future work to prove) that the problem of computing progress is PSPACE-hard. However, in general the size of the LTL formula is small, whereas the size of the search is huge. Hence, we expect our algorithm to be useful.

Providing a lower bound for the progress measure can be done in polynomial time. As we have shown, this bound is tight for invariants. Invariants form an important class of properties. Determining the class of LTL$_+$ formulae for which the bound is tight is another topic for further research.

The approach to handle the positive fragment of LTL seems not applicable to all of LTL. We believe that a different approach is needed and leave this for future research.

Acknowledgments We thank the referees for their constructive feedback.

References

[1] Robert B. Ash (1970): *Basic Probability Theory*. John Wiley & Sons.

[2] Christel Baier & Joost-Pieter Katoen (2008): *Principles of Model Checking*. The MIT Press.

[3] Patrick Billingsley (1995): *Probability and Measure*. John Wiley & Sons.

[4] Costas Courcoubetis & Mihalis Yannakakis (1995): *The Complexity of Probabilistic Verification*. Journal of the ACM 42(4), pp. 857–907, doi:10.1145/210332.210339.

[5] John G. Kemeny, J. Laurie Snell & Anthony W. Knapp (1966): *Denumerable Markov Chains*. Van Nostrand.

[6] Esteban Pavese, Victor Braberman & Sebastian Uchitel (2010): *My Model Checker Died!: how well did it do?* In: *Proceedings of the 2010 ICSE Workshop on Quantitative Stochastic Models in the Verification and Design of Software Systems*, ACM, Cape Town, pp. 33–40, doi:10.1145/1808877.1808884.

[7] Giuseppe Della Penna, Benedetto Intrigila, Igor Melatti, Enrico Tronci & Marisa Venturini Zilli (2006): *Finite Horizon Analysis of Markov Chains with the Murφ Verifier*. International Journal on Software Tools for Technology 8(4/5), pp. 397–409, doi:10.1007/s10009-005-0216-7.

[8] Wilson A. Sutherland (1975): *Introduction to Metric and Topological Spaces*. Clarendon Press.

[9] Willem Visser, Klaus Havelund, Guillaume Brat, SeungJoon Park & Flavio Lerda (2003): *Model Checking Programs*. Automated Software Engineering 10(2), pp. 203–232, doi:10.1023/A:1022920129859.

[10] Xin Zhang & Franck van Breugel (2010): *Model Checking Randomized Algorithms with Java PathFinder*. In: *Proceedings of 7th International Conference on Quantitative Evaluation of Systems*, IEEE, Williamburgh, pp. 157–158, doi:10.1109/QEST.2010.28.

[11] Xin Zhang & Franck van Breugel (2011): *A Progress Measure for Explicit-State Probabilistic Model-Checkers*. In Luca Aceto, Monika Henzinger & Jirí Sgall, editors: *Proceedings of the 38th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science* 6756, Springer-Verlag, Zurich, pp. 283–294, doi:10.1007/978-3-642-22012-8_22.