Verifying \( \omega \)-Regular Properties of Markov Chains

Doron Bustan\(^1\), Sasha Rubin\(^2\), and Moshe Y. Vardi\(^1\)

\(^1\) Rice University
\(^2\) The University of Auckland

Abstract. In this work we focus on model checking of probabilistic models. Probabilistic models are widely used to describe randomized protocols. A Markov chain induces a probability measure on sets of computations. The notion of correctness now becomes probabilistic. We solve here the general problem of linear-time probabilistic model checking with respect to \( \omega \)-regular specifications. As specification formalism, we use alternating Büchi infinite-word automata, which have emerged recently as a generic specification formalism for developing model checking algorithms. Thus, the problem we solve is: given a Markov chain \( M \) and automaton \( A \), check whether the probability induced by \( M \) of \( L(A) \) is one (or compute the probability precisely). We show that these problem can be solved within the same complexity bounds as model checking of Markov chains with respect to LTL formulas. Thus, the additional expressive power comes at no penalty.

1 Introduction

In model checking, we model a system as a transition system \( M \) and a specification as a temporal formula \( \psi \). Then, using formal methods, we check whether \( M \) satisfies \( \psi \) [7]. One of the most significant developments in this area is the discovery of algorithmic methods for verifying temporal logic properties of finite-state systems [25, 19, 6, 35]. This derives its significance both from the fact that many synchronization and communication protocols can be modelled as finite-state programs, as well as from the great ease of use of fully algorithmic methods. Looking at model-checking algorithms more closely, we can classify these algorithms according to two criteria. The first criterion is the type of model that we use – nondeterministic or probabilistic. The second criterion is the specification language.

For nondeterministic models and linear temporal logic (LTL), a close and fruitful connection with the theory of automata over infinite words has been developed [34–36]. The basic idea is to associate with each LTL formula a nondeterministic Büchi automaton over infinite words (NBW) that accepts exactly all the computations that satisfy the formula. This enables the reduction of various decision problems, such as satisfiability and model checking, to known automata-theoretic problems, yielding clean and asymptotically optimal algorithms. Furthermore, these reductions are very helpful for implementing temporal-logic based verification algorithms, cf. [14]. This connection to

\*\*\* Supported in part by NSF grants CCR-9988322, CCR-0124077, CCR-0311326, IIS-9908435, IIS-9978135, and EIA-0086264, by BSF grant 9800096, and by a grant from the Intel Corporation.
automata theory can also be extended to languages beyond LTL, such as ETL [36] and \( \mu \text{TL} \) [32].

In this paper we focus on model checking of probabilistic models. Probabilistic models are widely used to describe randomized protocols, which are often used in distributed protocols [5], communication protocols [8], robotics [29], and more. We use Markov chains as our probabilistic model, cf. [31]. A Markov chain induces a probability measure on sets of computations. The notion of correctness now becomes probabilistic: we say here that a program is correct if the probability that a computation satisfies the specification is one (we also discuss a quantitative notion of correctness, where we compute the probability that a computation satisfies the specification). Early approaches for probabilistic model checking of LTL formulas [20, 31] required determinization of NBW, which involves an additional exponential blow-up over the construction of automata from formulas [26], requiring exponential space complexity, unlike the polynomial space complexity of standard model-checking algorithms for LTL, cf. [30].

The exponential gap for probabilistic model checking was bridged in [9, 10], who provided a polynomial-space algorithm, matching the lower bound of [28]. The algorithm in [9, 10] is specialized to LTL (and ETL) specifications, and proceeds by induction on the structure of the formula. An automata-theoretic account of this algorithm was given in [11]. It is shown there that LTL formulas can be translated to a special type of NBW, which they call separated automata. (An NBW is separated if every two states that are located in the same strongly connected component have disjoint languages). As with the standard translation of LTL formulas to NBW, the translation to separated automata is exponential. It is then shown in [11] how to model check Markov chains, in nondeterministic logarithmic space, with respect to separated NBW as complemented specification. This yields a polynomial space upper bound for probabilistic model checking of LTL formulas.

The automata-theoretic framework in [11] is very specifically tailored to LTL. As mentioned earlier contrast, the automata-theoretic framework for model checking nondeterministic models is quite general and can also handle more expressive specification languages such as ETL and \( \mu \text{TL} \). This is not a mere theoretical issue. There has been a major recent emphasis on the development of industrial specification languages. These efforts resulted in a several languages [18, 23, 4, 2], culminating in an industrial standard, PSL 1.01 (www.accellera.com). Most of the new languages have the full power of NBW, i.e., they can express all \( \omega \)-regular languages. Thus, they are strictly more expressive than LTL [37], and, thus, not covered by the framework of [9–11].

In this paper we solve the general problem of probabilistic model checking with respect to \( \omega \)-regular specifications. As specification formalism, we use alternating Büchi infinite-word automata (ABW); see discussion below. Thus, the problem we solve is: Given a Markov chain \( M \) and an ABW \( A \), check whether the probability induced by \( M \) of \( L(A) \) is one (i.e., whether \( P_M(L(A)) = 1 \)). (A more refined problem is to calculate the probability precisely, see Appendix D.)

The motivation for using ABWs as a specification formalism is derived from recent developments in the area of linear specification languages. First, ABWs have been used as an intermediate formalism between LTL formulas and nondeterministic Büchi word automata (NBW). As shown in [12, 13], one can exploit the linear translation from LTL
formulas to ABWs ([33]) for an early minimization, before the exponential translation to NBW. Second, not only can logics such as ETL and $\mu$TL be easily translated to ABW, but also most of the new industrial languages can be translated to ABW. Furthermore, for some of them efficient such translations are known (cf. [1]). Thus, ABW can serve as a generic specification formalism for developing model checking algorithms. Note that applying the techniques of [9, 10] to ABW specifications requires converting them first to NBW at an exponential cost, which we succeed here in avoiding.

We present here an algorithm for model checking of Markov chains, using ABWs as specifications. The space complexity of the algorithm is polylogarithmic in $M$ and polynomial in $A$. The linear translation of LTL to ABW implies that this complexity matches the lower bound for this problem.

As in [10, 11], our algorithm uses the subset construction to capture the language of every subset of states of $A$ (an infinite word $w$ is in the language of a set $Q$ of states if $w \in L(s)$ for every state $s \in Q$ and $w \notin L(s)$ for every $s \notin Q$). While for LTL, a straightforward subset construction suffices, this is not the case for ABW. A key technical innovation of this paper is our use of two nondeterministic structures that correspond to the alternating automaton $A$ to capture the language of every set of automaton states. The first nondeterministic structure is an NBW $A_f$ called the full automaton, and the second a “slim” version of the full automaton without accepting conditions, which we call the local transition system $T_A$. Every state $q$ of $A_f$ and $T_A$ corresponds to a set $Q$ of states in $A$. While is possible, however, that a several states of $A_f$ correspond to the same set of states of $A$, every state of $T_A$ corresponds to a unique set of states of $A$. The model-checking algorithm make use of the products $G$ and $G_f$ of the Markov chain $M$ with $T_A$ and $A_f$, respectively.

2 Preliminaries

2.1 Automata

Definition 1. A nondeterministic Büchi word automaton (NBW) is $A = \langle \Sigma, S, S_0, \delta, F \rangle$, where $\Sigma$ is a finite alphabet, $S$ is a finite set of states, $\delta : S \times \Sigma \rightarrow 2^S$ is a transition function, $S_0 \subseteq S$ is a set of initial states, and $F \subseteq S$ is a set of accepting states.

Let $w = w_0, w_1, \ldots$ be an infinite word over $\Sigma$. For $i \in \mathbb{N}$, let $w^i = w_i, w_{i+1}, \ldots$ denote the suffix of $w$ from its $i$’th letter. A sequence $s = s_0, s_1, \ldots$ in $S^\omega$ is a run of $A$ over an infinite word $w \in \Sigma^\omega$, if $s_0 \in S_0$ and for every $i > 0$, we have $s_{i+1} \in \delta(s_i, w_i)$. We use $\inf(\rho)$ to denote the set of states that appear infinitely often in $\rho$. A run $\rho$ of $A$ is accepting if $\inf(\rho) \cap F \neq \emptyset$. An NBW $A$ accepts a word $w$ if $A$ has an accepting run over $w$. We use $L(A)$ to denote the set of words that are accepted by $A$. For $s \in S$, we denote by $A^{(s)}$ the automaton $A$ with a single initial state $s$. We write $L(s)$ (the language of $s$) for $L(A^{(s)})$ when $A$ is clear from the context.

Before we define an alternating Büchi word automaton, we need the following definition. For a given set $X$, let $B^+(X)$ be the set of positive Boolean formulas over $X$ (i.e., Boolean formulas built from elements in $X$ using $\land$ and $\lor$), where we also allow the formulas $\text{true}$ and $\text{false}$. Let $Y \subseteq X$. We say that $Y$ satisfies a formula $\theta \in B^+(X)$ if the truth assignment that assigns true to the members of $Y$ and assigns false to the members of $X \setminus Y$ satisfies $\theta$. 
Definition 2. A tree is a set $X \subseteq \mathbb{N}^+$, such that for $x \in \mathbb{N}^+$ and $n \in \mathbb{N}$, if $xn \in X$ then $x \in X$.

We call the node $\epsilon$ the root of the tree. A node $x$ is the parent of another node $t$ and $t$ is a child of $x$ if $t$ is of the form $xn$. The level of a node $x$, denoted $|x|$, is its distance from the root $\epsilon$; in particular, $|\epsilon| = 0$. A branch $\beta = x_0, x_1, \ldots$ of a tree is a maximal sequence of nodes such that $x_0$ is the root $\epsilon$ and $x_i$ is the parent of $x_{i+1}$ for all $0 < i < |\beta|$. Note that $\beta$ can be finite or infinite. A $\Sigma$-labelled tree, for a finite alphabet $\Sigma$, is a pair $(\tau, T)$, where $\tau$ is a tree and $T$ is a mapping from the nodes of $\tau$ to $\Sigma$ that assigns to every node of $\tau$ a label in $\Sigma$. We often consider $T$ as the labelled tree. A branch $\beta = x_0, x_1, \ldots$ of $\tau$ defines a word $T(\beta) = T(x_0), T(x_1), \ldots$ consisting of the sequence of labels along the branch.

Definition 3. An alternating Büchi word automaton (ABW) is $A = \langle \Sigma, S, s_0, \delta, F \rangle$, where $\Sigma$, $S$, and $F$ are as in NBW, $s_0 \in S$ is a single initial state, and $\delta : S \times \Sigma \rightarrow B^+ (S)$ is a transition function.

Because of the universal choice in alternating transitions, a run of an alternating automaton is a tree rather than a sequence. A run of a set $A$ on an infinite word $w = w_0, w_1, \ldots$ is a (possibly infinite) $S$-labelled tree $\tau$ such that $T(\epsilon) = s_0$ and the following holds:

- if $|x| = i$, $T(x) = s$, and $\delta(s, w_i) = \emptyset$, then $x$ has $k$ children $x \cdot 1, \ldots, x \cdot k$, for some $k \leq |S|$, and $\{T(x \cdot 1), \ldots, T(x \cdot k)\}$ satisfies $\emptyset$.

The run $\tau$ is accepting if every infinite branch in $\tau$ includes infinitely many labels in $F$. Note that the run can also have finite branches; if $|x| = i$, $\tau(x) = s$, and $\delta(s, w_i) = \emptyset$, then $x$ need not have children.

We define the projected graph of ABW $A = \langle \Sigma, S, s_0, F \rangle$ as a graph $(S, E)$ where $(s_1, s_2)$ is an edge in $E$ if $s_2$ appears in $\delta(s_1, \sigma)$ for some $\sigma \in \Sigma$.

Definition 4. An alternating weak word automaton (AWW) is an ABW $A = \langle \Sigma, S, s_0, F \rangle$ such that for every strongly connected component $C$ of the projected graph of $A$, either $C \subseteq F$ or $C \cap F = \emptyset$.

Lemma 1. [17] Let $A$ be an ABW. Then there exists an AWW $A_w$ such that $L(A) = L(A_w)$ and the size of $A_w$ is quadratic in the size of $A$. Furthermore, $A_w$ can be constructed in time quadratic in the size of $A$.

Given two AWW $A_1$ and $A_2$, we can construct AWW for $\Sigma^* \setminus L(A_1)$, $L(A_1) \cap L(A_2)$, and $L(A_1) \cup L(A_2)$, which are linear in their size, relative to $A_1$ and $A_2$ [24].

Next, we present a translation of ABWs into NBWs. Both the translation and its correctness are derived directly from [22]. We say that a tuple $(Q_1, Q_2) \subseteq 2^{S \times S}$ is consistent with respect to $A$ if $Q_2 \subseteq Q_1 \setminus F$.

Definition 5. Given an ABW $A = \langle \Sigma, S, s_0, F \rangle$, we define $N(A)$ as the NBW $A_n = \langle \Sigma, S_n, (\{s_0\}, \{s_0\} \setminus F), \delta_n, F_n \rangle$, such that:
Let $\Psi$ be measurable w.r.t. the sequence space $L$ and countable union. Consider a language $\Delta$ infinite sequences of states, $\Sigma$ space called the sequence space $M$ path of not change the complexity of verifying the Markov chain [10]. Note, that every infinite

states, $P$ in Markov chains

We model probabilistic systems by finite Markov chains. The basic intuition is that transitions between states are governed by some probability distribution.

Definition 6. A Markov chain is a tuple $M = \langle X, P_T, P_I \rangle$ such that $X$ is a set of states, $P_T : (X \times X) \rightarrow [0, 1]$ is a transition probability distribution that assigns to every transition $(x_1, x_2)$ its probability. $P_T$ satisfies that for every $x_1 \in X$ we have $\sum_{x_2 \in X} P_T(x_1, x_2) = 1$. $P_I : X \rightarrow [0, 1]$ is an initial probability distribution that satisfies $\sum_{x \in X} P_I(x) = 1$.

We denote by $M^{(x)}$ the Markov chain $M$ with $P_I$ that maps $x$ to 1. Sometimes we consider a Markov chain as a graph $\langle X, E \rangle$ where $(x_1, x_2) \in E$ iff $P_T(x_1, x_2) > 0$. For an alphabet $\Sigma = 2^{AP}$, let $V : X \rightarrow \Sigma$ be a labelling function, then each path $\rho$ in $X^*$ or in $X^\omega$ in $M$ is mapped by $V$ to a word $w$ in $\Sigma^*$ or in $\Sigma^\omega$ respectively. For simplicity we assume that $\Sigma = X$ and that $V(x) = x$ for every $x \in X$; this simplification does not change the complexity of verifying the Markov chain [10]. Note, that every infinite path of $M$ is a word in $X^\omega$ but the converse does not necessarily hold.

As in the standard theory of Markov processes (see [15, 16]), we define a probability space called the sequence space $\Psi_M = (\Omega, \Delta, P_M)$, where $\Omega = \Sigma^\omega$ is the set of all infinite sequences of states, $\Delta$ is the Borel field generated by the basic cylindric sets

$$\Delta(w_0, w_1, \ldots, w_n) = \{ w \in \Omega | \text{such that } w \text{ has prefix } w_0, w_1, \ldots, w_n \},$$

and $P_M$ is a probability distribution defined by

$$P_M(\Delta(w_0, w_1, \ldots, w_n)) = P_T(w_0) \cdot P_T(w_0, w_1) \cdot P_T(w_1, w_2) \cdots \cdot P_T(w_{n-1}, w_n).$$

($\Delta$ is the smallest set that contains the basic cylindric sets, and is closed under negation, and countable union.) Consider a language $L \subseteq \Sigma^\omega$, viewed as a specification, that is measurable w.r.t. the sequence space $\Psi_M$. We say that $M$ almost surely satisfies $L$ if $P_M(L) = 1$, that is, if “almost” all computations of $M$ are in $L$. In this paper we are concerned with the language of an ABW $A$, thus we want to determine whether $P_M(L(A)) = 1$. It is shown in [31] that $\omega$-regular languages are measurable.

The following property of Markov chains is called ergodicity and is proved in [16]. Let $M$ be a Markov chain, then a path of $M$, with probability one, enters a bottom
strongly connected component (BSCC) $K$ of $M$, and contains every finite path in $K$ infinitely often. In other words, let $L_e$ be the set of infinite words of $M$ such that every word $w$ in $L_e$ has a suffix that is contained in a BSCC $K$ of $M$, and contains every finite path in $K$ infinitely often. Then, $P_M(L_e) = 1$.

## 3 The Full Automaton and The Local Transition System

In this section we capture the behavior of an AWW $A$ using two nondeterministic systems. First we define the full automaton, which captures the languages of subsets of the states of $A$. Then we define the local transition system, which captures the local relations between subsets of states of $A$.

### 3.1 The Full Automaton

Given a AWW $A = \langle \Sigma, S, \delta, F \rangle$ (we ignore its initial state), we define its dual AWW $\hat{A} = \langle \Sigma, S, \hat{\delta}, \hat{F} \rangle$, where the Boolean formula $\hat{\delta}(s, \sigma)$ is obtained from $\delta(s, \sigma)$ by replacing every true with false and vice versa, and every $\vee$ with $\land$ and vice versa, in addition we define $\hat{F} = S \setminus F$. It is easy to see that $\hat{A}$ is an AWW.

**Lemma 2.** [24] Let $A$ be an AWW and $\hat{A}$ be its dual AWW. For every state $s$ we have that $\hat{L}(A(s)) = \Sigma^* \setminus L(\hat{A}(s))$.

Given an AWW $A$ and its dual AWW $\hat{A}$ we define the state space of the full automaton as a subset of $2^S \times 2^S \times 2^S \times 2^S$. We start with the following definition.

**Definition 7.** A tuple $(Q_1, Q_2, Q_3, Q_4)$ is consistent if $Q_2 = S \setminus Q_1$, $Q_3 \subseteq Q_1 \setminus F$, and $Q_4 \subseteq Q_2 \setminus \hat{F}$.

**Definition 8.** Given an AWW $A = \langle \Sigma, S, \delta, F \rangle$ we define its full automaton as the NBW $A_f = \langle \Sigma, \hat{S}_f, \hat{\delta}_f, \hat{F}_f \rangle$ where

- $\hat{S}_f$ is the set of consistent tuples over $2^S \times 2^S \times 2^S \times 2^S$.
- A state $(Q_1', Q_2', Q_3', Q_4')$ is in $\hat{\delta}_f((Q_1, Q_2, Q_3, Q_4), \sigma)$ if $Q_1' \models \land_{s \in Q_1} \delta(s, \sigma)$, $Q_2' \models \land_{s \in Q_2} \delta(s, \sigma)$, and either:
  1. $Q_3 = Q_4 = \emptyset$, $Q_5' = Q_5 \setminus F$, and $Q_6' = Q_6 \setminus \hat{F}$
  2. $Q_3 \neq \emptyset$ or $Q_4 \neq \emptyset$, there exists $Y_3 \subseteq Q_1'$ such that $Y_3 \models \land_{s \in Q_3} \hat{\delta}(s, \sigma)$ and $Y_3' = Y_3 \setminus F$, and there exists $Y_4 \subseteq Q_2'$ such that $Y_4 \models \land_{s \in Q_4} \hat{\delta}(s, \sigma)$ and $Y_4' = Y_4 \setminus \hat{F}$
- $\hat{F}_f = \{(Q_1, Q_2, Q_3, Q_4) \in \hat{S}_f | Q_3 = Q_4 = \emptyset\}$

The full automaton can be considered as an intersection of the two NBWs $N(A)$ and $N(\hat{A})$. The definition of the full automaton restricts the states of the intersection to consistent tuples.

**Theorem 2.** Let $A$ be an AWW and let $A_f$ be its full automaton, let $Q \subseteq S$ be a set of states, then for every state $(Q_1, Q_2, Q_3, Q_4)$ such that $Q_1 = Q$ we have that

$$\bigcap_{s \in Q} L(A(s)) \setminus \bigcup_{s \not\in Q} L(A(s)) = L(A_f(Q_1, Q_2, Q_3, Q_4))$$
Theorem 2 is proved in Appendix A.1. We now present more properties of the full automaton. We use these properties later.

**Definition 9.** Let \( A \) be an ABW and let \( w \) be an infinite word. We define the type of \( w \) w.r.t. \( A \) as the set type of \( A(w) = \{ s | A(s) \text{ accepts } w \} \).

The following lemma is a direct consequence of Theorem 2.

**Lemma 3.** Let \( A_f \) be full automaton, and let \((Q_1, Q_2, Q_3, Q_4)\) and \((Q_1', Q_2', Q_3', Q_4')\) be states of \( A_f \). Then,

- If \( Q_1 = Q_1' \) then \( L(A_f(Q_1, Q_2, Q_3, Q_4)) = L(A_f(Q_1', Q_2', Q_3', Q_4')) \).
- If \( Q_1 \neq Q_1' \) then \( L(A_f(Q_1, Q_2, Q_3, Q_4)) \cap L(A_f(Q_1', Q_2', Q_3', Q_4')) = \emptyset \).

Lemma 3 and Theorem 2 imply that the first element \( Q_1 \) of the states of \( A_f \) characterizes a distinct language.

**Definition 10.** The language of \( Q_1 \) is defined as \( L(Q_1) = L(A_f(Q_1, S \setminus Q_1, \emptyset, \emptyset)) \).

### 3.2 The local transition system

As observed above, it is sufficient to look at \( Q_1 \) in order to determine the language of a state of \( A_f \). Recall that \( L(Q_1) = L(A_f(Q_1, S \setminus Q_1, \emptyset, \emptyset)) \). We observe that if there exists a transition \((Q_1, Q_2, Q_3, Q_4), \sigma, (Q_1', Q_2', Q_3', Q_4')\) in \( \delta_f \), then for every state of the form \((Q_1, Q_2, Y_3, Y_4)\) there exists a state \((Q_1', Q_2', Y_3', Y_4')\) such that the transition \((Q_1, Q_2, Y_3, Y_4), \sigma, (Q_1', Q_2', Y_3', Y_4')\) is in \( \delta_f \).

These observations imply that there are some local relationships between the languages of the states of \( A_f \). Indeed, if a word \( w \) is in \( L((Q_1', Q_2', Q_3', Q_4')) \) then for the word \( \sigma \cdot w \) that is in \( L((Q_1, Q_2, Q_3, Q_4)) \), there exists a state of the form \((Q_1', Q_2', Y_3', Y_4')\) that is in \( \delta_f((Q_1, Q_2, Q_3, Q_4), \sigma) \). Thus, we can say that there exists a transition on \( \sigma \) from \( L(Q_1) \) to \( L(Q_1') \). The local transition system captures these relationships.

**Definition 11.** Given an AWW \( A = (\Sigma, S, \delta, F) \) we define its local transition system as \( T_A = (\Sigma, S_T, \delta_T) \) where

- \( S_T \) is the set of subsets of \( S \) and \( \delta_T \) is a function from \( S_T \) to \( 2^{S_T} \).
- A state \( Q' \) is in \( \delta_T(Q, \sigma) \) if \( Q' \models \wedge_{s \in Q} \delta(s, \sigma) \) and \( (S \setminus Q') \models \wedge_{s \notin Q} \hat{\delta}(s, \sigma) \).

**Example 2.** We now present an example of a full automaton and a local transition system. The example is presented at Figure 1. For simplicity we use a deterministic automaton \( A \). The figure shows \( A \)'s dual automaton \( \hat{A} \), the full automaton \( A_f \), and the local transition system \( T_A \). Note that \( \hat{A} \) has \( \hat{F} = S \), thus for every state \((Q_1, Q_2, Q_3, Q_4)\) of \( A_f \), we have \( Q_4 = \emptyset \). For this reason, and since \( Q_2 \) is always equal to \( S \setminus Q_1 \), we only write the sets \( Q_1 \) and \( Q_3 \) inside the states.

The definitions of the full automaton and the local transition system implies the following lemma:
Lemma 4. Let \( \mathcal{A} \) be an AWW and let \( \mathcal{A}_f \) and \( T_A \) be its full automaton and local transition system respectively. Let \((Q_1, Q_2, Q_3, Q_4)\) be a state of \( \mathcal{A}_f \), and let \( \sigma \) be a letter in \( \Sigma \). Then, for every state \( Q'_1 \) we have that \( Q'_1 \) is in \( \delta_T(Q_1, \sigma) \) iff there exists a state of the form \((Q'_1, Q'_2, Q'_3, Q'_4)\) in \( \delta_f((Q_1, Q_2, Q_3, Q_4), \sigma) \).

The proof of Lemma 4 is straightforward from the definitions of \( \mathcal{A}_f \) and \( T_A \). In particular for every state \((Q_1, Q_2, Q_3, Q_4)\) and infinite word \( w \), we have that \( \mathcal{A}^{(Q_1, Q_2, Q_3, Q_4)} \) has a run on \( w \) iff \( T_A^{(Q_1)} \) has a run on \( w \). However, we do not define accepting conditions for the local transition system. Thus, it is possible that \( T_A^{(Q_1)} \) has a run on \( w \), but \( \mathcal{A}_f^{(Q_1, Q_2, Q_3, Q_4)} \) does not have an accepting run on \( w \).

Lemma 5. Let \( T_A \) be a local transition system, and let \( Q, Q' \) and \( Q'' \) be states of \( T_A \). Let \( \sigma \) be a letter in \( \Sigma \). If \( Q'' \in \delta_T(Q', \sigma) \) and \( Q'' \in \delta_T(Q, \sigma) \), then \( Q = Q' \).

Lemma 5 is proved in Appendix A.2. When a transition system satisfies the property shown in Lemma 5, we say that the transition system is reverse deterministic.

4 Verifying Markov Chains

In this section we construct a product \( G_{\mathcal{M}, \mathcal{A}} \) of the Markov chain \( \mathcal{M} \) and the local transition system \( T_A \). We show that the problem of checking whether \( P_M(L(A)) = 1 \), can be reduced to checking for a state \((x, Q)\) of \( G \) whether the probability of \( L(Q) \cap x \cdot \Sigma^\omega \) is positive. Then, we show how to use the full automaton to solve this problem.
Definition 12. Let $A$ be an AWW, $T_A$ be $A$’s local transition system, and $M$ be a Markov chain. We define the graph $G_{M,A}$ as having vertex set $(x, Q)$ such that $x$ is a state of $M$ and $Q$ is a state of $T_A$. An edge $(x, Q) \rightarrow (x', Q')$ is included in $G_{M,A}$ if $M$ has a transition $x \rightarrow x'$ and $(Q, x, Q')$ is a transition in $T_A$.

When $A$ and $M$ are clear from the context, we write $G$ instead of $G_{M,A}$. Lemma 5 implies that for every three states $(x, Q)$, $(x', Q')$, and $(x'', Q'')$, if there is a transition from $(x, Q)$ to $(x'', Q'')$ and there is a transition from $(x', Q')$ to $(x'', Q'')$, then $x \neq x'$. We say that $G$ is reverse deterministic.

Example 3. We present in Figure 2 two Markov chains $M_1$ and $M_2$. We assume that the initial probability for each state in the Markov chains is $\frac{1}{2}$. The figure also presents the products $G_1$ and $G_2$ of $M_1$ and $M_2$ respectively, with the local transition system $T_A$ from Example 2.

![Fig. 2. Two Markov chains $M_1$ and $M_2$, and the graphs they impose $G_1$ and $G_2$.](image)

Every infinite path in $G$ projected on the first component gives a path in $M$. Conversely, every path of $M$ is the projection of at least one path in $G$. In fact, let $w = x_0 \cdot x_1 \cdot \ldots$ be a path of $M$. For each $j$, let $Q_j$ be the type of the suffix $x_j \cdot x_{j+1} \cdot x_{j+2} \cdot \ldots$. Then for each $j$ there is a transition $(Q_j, x_j, Q_{j+1})$ in $\delta_T$ and thus an edge $(x_j, Q_j) \rightarrow (x_{j+1}, Q_{j+1})$ in $G$. We call this path the augmented path corresponding to $w$.

Definition 13. For a state $(x, Q)$ of $G$ we denote by $P(x, Q)$ the probability that a path that starts in state $x$ has type $Q$, namely $P(x, Q) = P_M(x)$ has type $Q$. We call $(x, Q)$ probable if $P(x, Q) > 0$.

The importance of the probable states is demonstrated in the following two lemmas, which are proved in Appendix B.

Lemma 6. $P_M(L(Q)) = \sum_{x \in X} P_I(x) \cdot P(x, Q)$. 

Let $x$ be a state of a Markov chain with $P_I(x) > 0$. Then for every state $(x, Q)$ we have that if $(x, Q)$ is probable, then $P_M(L(Q)) > 0$. Thus we conclude Lemma 7.

**Lemma 7.** Let $s$ be a state of an AWWW $A$ and let $M$ be a Markov chain. Then, $P_M(L(s)) < 1$ iff there exists a state $x$ of $M$ and a set $Q \subseteq S$ such that $P_I(x) > 0$, $s \notin Q$ and the state $(x, Q)$ is probable.

Thus, in order to determine whether $P_M(L(s_0)) = 1$, it is enough to determine the probable states of $G$. In the rest of this section we show how to identify the probable states. We define $H$ as the restriction of $G$ to the probable states.

**Example 4.** Look again at Figure 2. It is easy to see that given that a path of $(M)$ path of states. We define $H$ iff there exists a state $x$ of $M$ and a set $Q \subseteq S$ such that $P_I(x) > 0$, $s \notin Q$ and the state $(x, Q)$ is probable.

This implies that in $G_1$, $H$ is the subgraph induced by the states $(a, \{1, 2\})$ and $(b, \{1\})$. On the other hand, looking at $M_2$, we see that a path that starts at $a$ is of the form $(ab)^\omega$ and a path that starts at $b$ is of the form $(ba)^\omega$. Thus, in $G_2$, $H$ is the subgraph induced by the states $(a, \{1\})$ and $(b, \emptyset)$.

We start with the following observation. Partition the language $L(Q)$ according to the first letter of a word and the type of the suffix that starts from the second letter. Then, for every state $(x, Q)$ of $G$ we have $P(x, Q) = \sum_{(x', Q') \rightarrow (x, Q)} P_T(x, x') \cdot P(x', Q')$. Note that if $(x, Q) \rightarrow (x', Q')$ is an edge of $G$, then $P_T(x, x') > 0$ and thus $P(x, Q) \geq P_T(x, x') \cdot P(x', Q')$. Hence, if $(x', Q')$ is probable then all its ancestors are probable. This implies that it is sufficient to identify the BSCCs of $H$ and then to construct the set of probable states using backward reachability.

Let $C$ be an SCC of $G$. If it contains some probable state $(x, Q)$, then since all the states in $C$ are ancestors of $(x, Q)$, all states in $C$ are probable. That is, either $C$ is an SCC of $H$ or $C \cap H = \emptyset$. Recall that every path in $C$ projects to a path in $M$. So the set of first components of all members of $C$ are in the same SCC, say $K$, of $M$, which is the SCC of $M$ containing $x$. We say that $C$ corresponds to $K$. Note that distinct SCC’s of $G$ may correspond to the same $K$.

Theorem 3 characterizes the SCCs of $G$ which are the BSCCs of $H$. Before we present the theorem we need the following notation. For a tuple $E = (E_1, E_2, \ldots, E_n)$, we define $\pi_i(E) = E_i$ to be the $i$’th element in $E$. This notation is extended naturally to sequences of tuples.

**Definition 14.** A finite path $\rho_G$ in $G$ is fulfilling if there exists a path $\rho_I$ in $A_I$ such that $\pi_2(\rho_G) = \pi_1(\rho_I)$, the first state of $\rho_I$ is of the form $(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus \hat{F}))$, and the last state of $\rho_I$ is of the form $(Q_1', Q_2', \emptyset, \emptyset)$.

**Theorem 3.** Let $C$ be an SCC of $G$. Then $C$ is a BSCC of $H$ iff it satisfies the following conditions:

1. $C$ corresponds to a BSCC $K$ of $M$.
2. Every finite path of $K$ is a projection of a path in $C$.
3. $C$ contains a fulfilling path.

The proof of Theorem 3 is highly nontrivial and is presented at Appendix B.1.
5 Algorithms

In Figure 3 we present the algorithm that determines for an AWW \( A \) and a Markov chain \( M \) whether \( \Pr_M(L(A)) = 1 \). An extension for exact probability is presented in Appendix D. Theorem 3 implies that the algorithm mark the BSCCs of \( H \). Thus, \( B \) is the set of probable states. Lemma 7 implies that the algorithm returns true iff \( \Pr_M(L(A)) = 1 \). Finding SCCs in \( G \) that correspond to BSCCs in \( M \), and doing backward reachability can be done in time linear in the size of \( G \). The most complex part of the algorithm is to identify, SCCs \( C \) of \( G \) that satisfy:

1. \( C \) corresponds to a BSCC \( K \) of \( M \).
2. Every finite path of \( K \) is a projection of a path in \( C \).
3. \( C \) contains a fulfilling path.

Construct the set \( B \) of all states of \( G \) from which the marked SCCs are reachable.

\[
\text{return true\; iff\; for\; every\; state\; } (x, Q) \in B, \; \text{if } P_I(x) > 0, \; \text{then } s_0 \in Q.
\]

**Fig. 3.** The model-checking algorithm.

The following lemma is proved in [10]. The only property of \( G \) that they use is that \( G \) is reverse deterministic.

**Lemma 8.** Let \( C \) be an SCC of \( G \) that corresponds to an SCC \( K \) of \( M \). Then the following are equivalent:

1. Every finite path in \( K \) is a projection of a path in \( C \).
2. No other SCC of \( G \) corresponding to \( K \) is an ancestor of \( C \).

Lemma 8 implies that the second task is equivalent to checking whether there is no ancestor SCC of \( C \) that corresponds to \( K \). This check can be easily done while scanning the SCCs of \( G \).

**Example 5.** In \( G_1 \) at Figure 2 there are two SCC’ s that correspond to the single BSCC of \( M_1 \). The SCC of \( \{a, \{1, 2\}\} \) and \( \{b, \{1\}\} \) does not have ancestors and contains the fulfilling path \( (a, \{1, 2\}), (a, \{1, 2\}), (a, \{1, 2\}) \) that corresponds to the path \( (\{1, 2\}, \{1\}), (\{1, 2\}, \{1\}) \) in \( A_f \), thus, this SCC is the BSCC of \( H \). In \( G_2 \) there are two SCCs that correspond to the single BSCC of \( M_2 \) and neither of them have an ancestor. However, only the SCC of \( \{a, \{1\}\} \) and \( \{b, \emptyset\} \) has a fulfilling path, thus it is the BSCC of \( H \).
We now explain how to check whether an SCC \( C \) of \( G \) contains a fulfilling path. We construct the product \( G_f = \mathcal{M} \times \mathcal{A}_f \), similarly to the construction of \( G \).

**Definition 15.** Let \( \mathcal{A} \) be an AWW, \( \mathcal{A}_f \) be \( \mathcal{A} \)’s full automaton, and \( \mathcal{M} \) be a Markov chain. We define the full graph \( G_f \) as having vertex set \((x, (Q_1, Q_2, Q_3, Q_4))\) such that \( x \) is a state of \( \mathcal{M} \) and \((Q_1, Q_2, Q_3, Q_4)\) is a state of \( \mathcal{A}_f \). An edge \((x, (Q_1, Q_2, Q_3, Q_4)) \rightarrow (x', (Q'_1, Q'_2, Q'_3, Q'_4))\) is included in \( G_f \) if \( \mathcal{M} \) has a transition \( x \rightarrow x' \) and \((Q'_1, Q'_2, Q'_3, Q'_4)\) is in \( \delta_f((Q_1, Q_2, Q_3, Q_4), x) \).

**Lemma 9.** An SCC \( C \) of \( G \) contains a fulfilling path iff there exists a path \((x_0, (Q^0_1, Q^0_2, Q^0_3, Q^0_4)), (x_1, (Q^1_1, Q^1_2, Q^1_3, Q^1_4)), \ldots, (x_n, (Q^n_1, Q^n_2, Q^n_3, Q^n_4))\) in \( G_f \) such that the path \((x_0, Q^0_1), (x_1, Q^1_1), \ldots, (x_n, Q^n_1)\) is contained in \( C \), \( Q^0_3 = Q^n_3 \setminus F \), and \( Q^0_4 = Q^n_4 \setminus F \).

Lemma 9 is proved in Appendix C.

**Complexity** Finding SCCs in \( G \) that correspond to BSCCs in \( \mathcal{M} \), and doing backward reachability can be done in linear time and polylogarithmic space in \( |G| \). Constructing BSCCs of \( \mathcal{M} \) can be done in time linear in \( |\mathcal{M}| \), identifying SCCs of \( G \) that correspond to these BSCCC can be done in time linear in \( |G| \). Marking SCCs that do not have ancestors that correspond to the same BSCC in \( \mathcal{M} \) can also be done in time linear in \( |G| \). Checking that an SCC of \( G \) contains a fulfilling path can be done in time linear in \( |G_f| \), simply by scanning \( G_f \) and \( G \) in parallel, thus, the algorithm can be implemented in time linear in \( |\mathcal{M} \times \mathcal{A}_f| \). Since the size of \( \mathcal{A}_f \) is \( 2^{O(|\mathcal{A}|)} \), we have that the time complexity of the algorithm is \( |\mathcal{M}| \cdot 2^{O(|\mathcal{A}|)} \).

As for space complexity we show that algorithm works in space polynomial in \( |\mathcal{A}| \) and polylogarithmic in \( |\mathcal{M}| \). We rewrite the conditions of Theorem 3, Lemma 7, and Lemma 8 as follows: \( P_{\mathcal{M}}(L(\mathcal{A})) < 1 \) iff there exists a probable state \((x_0, Q_0)\) such that \( s_0 \not\in Q \) and \( P_f(x_0) > 0 \). This is true iff \((x_0, Q_0)\) reaches a state \((x, Q)\) that is in a BSCC of \( H \), \( s_0 \not\in Q_0 \), and \( P_f(x_0) > 0 \). That is

1. \((x, Q)\) is reachable from a state \((x_0, Q_0)\) such that \( P_f(x_0) > 0 \) and \( s_0 \not\in Q_0 \).
2. \( x \) is in a BSCC of \( \mathcal{M} \) (Theorem 3, (1)). This condition is equivalent to the following: for every state \( x' \) of \( \mathcal{M} \) we have that if there exists a path in \( \mathcal{M} \) from \( x \) to \( x' \) then there exists a path from \( x' \) to \( x \).
3. No other SCC of \( G \) that corresponds to the SCC of \( x \) in \( \mathcal{M} \) is the ancestor of the SCC of \((x, Q)\) (Lemma 8). This condition is equivalent to the following: for every state \((x', Q')\), if there exists a path from \((x', Q')\) to \((x, Q)\), then either there exists a path from \((x, Q)\) to \((x', Q')\), or there is no path from \( x \) to \( x' \).
4. The SCC of \((x, Q)\) contains a fulfilling path (Theorem 3, (3)). By Lemma 9 this condition is equivalent to the following: there exists a path in \( G_f \) from a state \((x'_1, (Q'_1, Q'_2, Q'_1 \setminus F, Q'_2 \setminus F))\) to a state \((x'', (Q''_1, Q''_2, Q''_2, Q''_1 \setminus F))\) such that the projection of the path on \( G \) is contained in the SCC of \((x, Q)\). This condition is equivalent to: there is a path from a state \((x', (Q'_1, Q'_2, Q'_1 \setminus F, Q'_2 \setminus F))\) to a state \((x'', (Q''_1, Q''_2, Q''_2, Q''_1 \setminus F))\) in \( G_f \) and there are paths from \((x, Q)\) to \((x', Q')\), from \((x'', Q''_1)\) to \((x', Q')\), and from \((x', Q')\) to \((x, Q)\) in \( G \).
In [27] it is shown that checking whether there is a path from one state to another in a graph with \( n \) states requires \( \log^2(n) \) space. This implies that the conditions above can be checked in space \( O(\log^2(|G_f|)) = \log^2(|M| \cdot 2^{O(|A|)}) = O(\log^2(|M|) + \log(|M|) \cdot |A| + |A|^2) = O(\log^2(|M|) + |A|^2) \).

6 Concluding remarks

We presented here an optimal solution to the general problem of linear-time probabilistic model checking with respect to \( \omega \)-regular specifications, expressed by alternating automata. Beyond the interest in the problem itself, our solution is interesting from a theoretical perspective, since the concept of full automaton may have other applications. More work is needed in reducing our result to practice. One direction is to extend the ProbaTaf system, which currently handles LTL specifications of Markov chain [11], to alternating automata specifications. Another, is to combine the symbolic approach to alternating automata [21] with the symbolic approach to probabilistic model checking [3].

References


A Proofs for Section 3

A.1 Proof of Theorem 2

Lemma 10. Let $A$ be an AWW, let $s$ be a state in $S$, and let $\sigma$ be a letter in $\Sigma$. Then, for every $Q \subseteq S$, we have that $Q \models \delta(s, \sigma)$ if $S \setminus Q \models \delta(s, \sigma)$.

Lemma 10 can be proved by induction on the structure of the formula $\delta(s, \sigma)$.

Lemma 11. Let $A$ be an AWW, let $w$ be an infinite word, and let $Q$ be a subset of $S$ s.t. $w$ is in $\bigcap_{s \in Q} L(A^{(s)}) \bigcap_{s \notin Q} L(A^{(s)})$. Then, $Q = \text{type}_A(w)$.

Proof. Since $w \in \bigcap_{s \in Q} L(A^{(s)})$, we have $Q \subseteq \text{type}_A(w)$. Since $w \in \bigcap_{s \notin Q} L(A^{(s)})$, we have $\text{type}_A(w) \subseteq Q$, thus, $Q = \text{type}_A(w)$.  

Lemma 12. Let $A$ be an AWW, let $A'$ be its dual, and let $w$ be an infinite word. Then $S \setminus \text{type}_A(w) = \text{type}_{A'}(w)$.

Proof. Let $s$ be a state, then $s \in S \setminus \text{type}_A(w)$ iff $w \notin L(A^{(s)})$. By Lemma 2, $w \notin L(A^{(s)})$ iff $w \in L(A^{(s)})$ iff $s \in \text{type}_A(w)$.  

Lemma 13. Let $A$ be an AWW and let $w$ be an infinite word. Then for every $i$ and every $s \in \text{type}_A(w^i)$, we have $\text{type}_A(w^{i+1}) \models \delta(s, w_i)$

Proof. Let $s$ be a state in $\text{type}_A(w^i)$, then $A^{(s)}$ accepts $w^i$. Let $Q'$ be the set of successors of $s$ in an accepting run tree of $A^{(s)}$ on $w^i$. Then, for every $t \in Q'$ we have that $A^{(t)}$ accepts $w^{i+1}$, thus $Q' \subseteq \text{type}_A(w^{i+1})$. Since, $Q' \models \delta(s, w_i)$, we have $\text{type}_A(w^{i+1}) \models \delta(s, w_i)$.  

Definition 16. Let $A$ be an AWW and $A_f$ be the full automaton of $A$. Let $w$ be an infinite word. We define $\text{Max}_A(w)$ to be the state $(\text{type}_A(w), \text{type}_A(w), \text{type}_A(w) \setminus F, \text{type}_A(w) \setminus F)_{A_f}$ of $A_f$.

Lemma 12 implies that $\text{Max}_A(w)$ is a consistent tuple.

Lemma 14. Let $w$ be an infinite word and let $A$ be an AWW. Then there exists a run of $A_f$ on a prefix of $w$ that starts at $\text{Max}_A(w)$ and that reaches a state in $F_f$.

Proof. Let $(Q^0_1, Q^0_2, Q^0_3, Q^0_4) = \text{Max}_A(w)$. We define two sets of trees, $\Phi_3$ is a set of $|Q^0_1|$ many run trees s.t. for every $s \in Q^0_1$ there exists a unique tree in $\Phi_3$ that is an accepting run tree of $A^{(s)}$ on $w$. Similarly, we define $\Phi_4$ as a set of $|Q^0_2|$ many run trees s.t. for every $t \notin Q^0_1$ there exists a unique tree in $\Phi_4$ that is an accepting run tree of $A^{(t)}$ on $w$. We define $\Phi = \Phi_3 \cup \Phi_4$. Note that every path of a tree in $\Phi_3$ contains infinitely many accepting states in $F$, and every path of a tree in $\Phi_4$ contains infinitely many accepting states in $\bar{F}$. Furthermore, the branching degree of every node in these trees is bounded by $|S|$. By Konigs lemma, there exists an index $i$ s.t. for every tree in $\Phi$ and every path $\xi$ in the tree there exists $k \leq i$ s.t. $\xi_k$ is an accepting state.

We construct a run of $A_f^{(Q^0_1, Q^0_2, Q^0_3, Q^0_4)}$ on $w$ inductively, until it reaches a state in $F_f$. The run satisfies the following properties:
1. For every $k \leq i$ we have that if a state $s$ is in $Q_k^i$, then there exists a path $\xi$ in a tree in $\Phi_3$ s.t. $\xi_0, \xi_1, \ldots, \xi_k$ does not contain states in $F$ and $\xi_k = s$.

2. For every $k \leq i$ we have that if a state $\ell$ is in $Q_k^i$, then there exists a path $\xi$ in a tree in $\Phi_3$ s.t. $\xi_0, \xi_1, \ldots, \xi_k$ does not contain states in $F$ and $\xi_k = \ell$.

The first state of the run is $(Q_1^0, Q_2^0, Q_3^0, Q_4^0)$. It is easy to see that the properties above hold for $k = 0$. Suppose that for some $k < i$, we have a prefix of a run of $A_f$ that satisfies the properties above. We define $Q_{k+1}^i = type_A(w^{k+1})$, $Q_{k+1}^i = S \setminus Q_k^{k+1}$. As for $Q_{k+1}^i$ and $Q_{k+1}^{k+1}$, the induction hypothesis implies that for every state $s$ in $Q_{k+1}^{k+1}$ there exists a path $\xi^s$ in a tree in $\Phi_3$ s.t. $\xi_0^s, \xi_1^s, \ldots, \xi_k^s$ does not contain states in $F$ and $\xi_k^s = s$. Let $Y_s^s$ be the set of successors of $\xi_s^s$ in the tree. We define $Y_s = \bigcup_{s \in Q_s^k} Y_s^s$, and $Q_{k+1}^s = Y_s \setminus F$. Since for every $s$ in $Q_{k+1}^s$ we have that the successors of $\xi_s^s$ in the run tree accepts $w^{k+1}$, we have that $Y_s \subseteq type_A(w^{k+1})$, thus, $Y_s \subseteq Q_{k+1}^s$. In a similar way we define $Y_4 \subseteq Q_4^{k+1}$ using trees of $\Phi_4$, and define $Q_{k+1}^s = Y_s \setminus F$. Since $Y_s \subseteq Q_{k+1}^s$ and $Y_4 \subseteq Q_4^{k+1}$, the state $(Q_{k+1}^s, Q_{k+1}^s, Q_{k+1}^{k+1}, Q_{k+1}^{k+1})$ is consistent.

Obviously, the properties above holds for $k+1$. We prove that $(Q_{k+1}^s, Q_{k+1}^s, Q_{k+1}^s, Q_{k+1}^s)$ is in $\delta_f((Q_k^s, Q_k^s, Q_k^s, Q_k^s), w_k)$. Lemma 13 implies that for every state $s \in Q_k^s$ we have $Q_{k+1}^s = \delta(s, w_k)$. Lemma 12 implies that $Q_{k+1}^s = type_A(w^{k+1})$. Lemma 13 implies that for every $t \in Q_{k+1}^s$, we have $Q_{k+1}^s = \delta(t, w_k)$. The definition of $Y_3$ implies that for every state $s$ in $Q_{k+1}^s$ we have $Y_3 = \delta(s, w_k)$. Similarly, for every $t \in Q_4^s$ we have $Y_4 = \delta(t, w_k)$. This completes the proof that $(Q_{k+1}^s, Q_{k+1}^s, Q_{k+1}^s, Q_{k+1}^s)$ is in $\delta_f((Q_k^s, Q_k^s, Q_k^s, Q_k^s), w_k)$.

Since every path in a tree in $\Phi$ contains an accepting state before the $i$'th state, the properties above imply that the run reaches an accepting state before the $i$'th state. □

**Lemma 15.** Let $A$ be a AWW and let $A_f$ be its full automaton, let $Q \subseteq S$ be a set of states, then for every state $Q(1, Q_2, Q_3, Q_4)$ s.t. $Q_1 = Q$ we have that

$$\bigcap_{s \in Q} L(A(s)) \subseteq L(A_f(Q_1, Q_2, Q_3, Q_4))$$

**Proof.** We prove that every $w$ in $\bigcap_{s \in Q} L(A(s)) \subseteq L(A_f(Q_1, Q_2, Q_3, Q_4))$ is also in $L(A_f(Q_1, Q_2, Q_3, Q_4))$.

We construct the run of $A_f$ inductively in finite sequences, s.t. for every sequence, the last state $(Q_1^i, Q_2^i, Q_3^i, Q_4^i)$ in the sequence is $Max_A(w^i)$. Moreover, every sequence except for the first, contains an accepting state. Since, the run contains infinitely many sequences, the run is accepting.

- **Base:** Lemma 11 implies that $Q_1 = type_A(w)$. Lemma 19 implies that there exists a transition from $(Q_1, Q_2, Q_3, Q_4)$ to $Max_A(w^1)$ on $w_0$. This is the first sequence.

- **Induction step:** Suppose that the $i$'th sequence ends at state $(Q_1^i, Q_2^i, Q_3^i, Q_4^i)$ s.t. $(Q_1^i, Q_2^i, Q_3^i, Q_4^i) = Max_A(w^i)$. Then, Lemma 14 implies that there exists a run of $A_f$ that starts from $(Q_1^i, Q_2^i, Q_3^i, Q_4^i)$ and reaches an accepting state $(Q_1^j, Q_2^j, \emptyset, \emptyset)$ for some $j > k$. The definition of $A_f$ implies that there exists a transition from $(Q_1^j, Q_2^j, \emptyset, \emptyset)$ to $Max_A(w^{j+1})$ on $w_j$. We take $(Q_1^{j+1}, Q_2^{j+1}, Q_3^{j+1}, Q_4^{j+1}), \ldots, (Q_1^i, Q_2^i, \emptyset, \emptyset), Max_A(w^{j+1})$ to be the $i + 1$'th sequence. □
Lemma 16. Let $A$ be a AWW and let $A_f$ be its full automaton, let $Q \subseteq S$ be a set of states, then for every state $(Q_1, Q_2, Q_3, Q_4)$ s.t. $Q_1 = Q$ we have that $\bigcap_{s \in Q} L(A^{(s)}) \cap \bigcap_{s \notin Q} L(A^{(s)}) \supseteq L\left(A_f^{(Q_1, Q_2, Q_3, Q_4)}\right)$

Proof. We prove that every $w$ in $L\left(A_f^{(Q_1, Q_2, Q_3, Q_4)}\right)$ is also in $\bigcap_{s \in Q} L(A^{(s)}) \cap \bigcap_{s \notin Q} L(A^{(s)})$. Let $\rho_f = (Q_1^0, Q_2^0, Q_3^0, Q_4^0), (Q_1^1, Q_2^1, Q_3^1, Q_4^1), \ldots$ be an accepting run of $A_f^{(Q_1, Q_2, Q_3, Q_4)}$ on $w$ and let $s$ be a state in $Q_1^t$. We need to show that $w$ is in $A^{(s)}$. Let $A_n = N(A)$ as defined in Definition 5. We prove that $A_n^{(Q_1, Q_2, Q_3, Q_4)}$ has an accepting run on $w$, by Theorem 1 this implies that $w$ is in $L(A^{(s)})$. Similarly, we can prove that $w$ is in $L(\bar{A}^{(s)})$ for every state $t$ in $Q_2^0$.

Let $i_0 = -1$ and let $i_1, i_2, \ldots$ be an infinite sequence of indices s.t. for every $j \geq 1$, the $i_j$’th state of $\rho$ is in $F$. We construct a run $\rho_n = (Y_1^0, Y_2^0), (Y_1^1, Y_2^1), \ldots$ s.t. For every $j$ and $i < k \leq i_{j+1}$ we define $Y_1^k = Q_1^{i_j}$, and

$$Y_2^k = \begin{cases} Y_1^k \setminus F & \text{if } Y_2^l = \emptyset \text{ for some } i_j < l < k \\ Q_3^k & \text{otherwise} \end{cases}$$

We prove that $\rho_n$ is an accepting run of $A_n$ on $w$. We prove by induction on $j$ that for every $j \geq 0$ we have that

1. $Y_1^{i_j+1} = Q_1^{i_j+1}$ and $Y_2^{i_j+1} = Q_2^{i_j+1}$.
2. For every $i_j < k \leq i_{j+1}$, there exists a transition from $(Y_1^k, Y_2^k)$ to $(Y_1^{k+1}, Y_2^{k+1})$ on $w_k$.
3. There exists $i_j < k \leq i_{j+1}$ s.t. $Y_2^k = \emptyset$

Since for every $j \geq 0$ there exists $k > i_j$ s.t. $Y_2^k = \emptyset$, the run is accepting.

We start with two observations. The definitions of $A_f$ and $A_n$ imply the following observation.

Observation 17 For a state $(Q_1, Q_2, Q_3, Q_4)$ of $A_f$ s.t. $Q_3 \neq \emptyset$, a transition from $(Q_1, Q_2, Q_3, Q_4)$ to $(Q'_1, Q'_2, Q'_3, Q'_4)$ on $\sigma$ in $A_f$ implies a transition from $(Q_1, Q_3)$ to $(Q'_1, Q'_3)$ in $A_n$.

Recall that if there exists a transition in $A_f$ from $(Q_1, Q_2, Q_3, Q_4)$ to $(Q'_1, Q'_2, Q'_3, Q'_4)$ on $\sigma$, then there exists a transition in $A_f$ from $(Q_1, Q_2, Q_3, Q_4)$ to $(Q'_1, Q'_2, Q'_3, Q'_4)$ on $\sigma$. This implies that:

Observation 18 If there exists a transition in $A_f$ from $(Q_1, Q_2, Q_3, Q_4)$ to $(Q'_1, Q'_2, Q'_3, Q'_4)$ on $\sigma$, then there exist a transition in $A_n$ from $(Q_1, Q_3)$ to $(Q'_1, Q'_3, F)$ on $\sigma$.

Next, we prove the induction claim.

- Base case $j = 0$: The definition of $\rho_n$ implies property 1. Let $l$ be the smallest index s.t. $i_j < l \leq i_{j+1}$ and $Q_3^l = \emptyset$. Then Observation 17 implies that for every $i_j < k < l$ property 2 holds. By the definition of $A_n$, property 3 holds in index $k = l$. Observation 18 implies that property 2 holds for $l \leq k \leq i_{j+1}$. 
Lemma 16 completes the proof of Theorem 2.

A.2 Proofs for the rest of Section 3

Lemma 19. Let A be an AWN, let \( A_f \) be its full automaton, and let \( w \) be an infinite word. Then, there exist a transition in \( A_f \) from every state of the form \((\text{type}_A(w), S \setminus \text{type}_A(w)), Q_3, Q_4)\) to \( \text{Max}_A(w^1) \) on \( w_0 \).

Proof. By Lemma 12, we have \( S \setminus \text{type}_A(w) = \text{type}_A(w) \). Lemma 13 implies that for every state \( s \) in \( \text{type}_A(w) \), we have \( \text{type}_A(w^1) \models \delta(s, w_0) \), and that for every state \( t \) in \( \text{type}_A(w) \), we have \( \text{type}_A(w^1) \models \hat{\delta}(t, w_0) \). If \( Q_3 = Q_4 = \emptyset \), then the definition of \( \delta \) implies the lemma directly. Otherwise, since \((\text{type}_A(w), S \setminus \text{type}_A(w), Q_3, Q_4)\) is consistent, we have that \( Q_3 \subseteq \text{type}_A(w) \setminus \emptyset \), and \( Q_4 \subseteq \text{type}_A(w) \setminus \emptyset \). We take \( Y_3 = \text{type}_A(w^1) \) and \( Y_4 = \text{type}_A(w^1) \). These sets satisfy the conditions of \( \delta \). \( \square \)

Theorem 2 implies the following lemma.

Lemma 20. For an AWN \( A \) and set of state \( Q \), we have \( L(Q) = \{ w \mid \text{type}_A(w) = Q \} \).

Lemma 5 Let \( T_A \) be a local transition system, and let \( Q, Q' \) and \( Q'' \) be states of \( T_A \). Let \( \sigma \) be a letter in \( \Sigma \). If \( Q'' \in \delta_T(Q', \sigma) \) and \( Q'' \in \delta_T(Q, \sigma) \), then \( Q = Q' \).

Proof. Assume to the contrary that the lemma does not hold, meaning, \( Q'' \in \delta_T(Q', \sigma) \) and \( Q'' \in \delta_T(Q, \sigma) \), but \( Q \neq Q' \). Then, w.l.o.g. there exists a state \( s \) in \( Q \) that is not in \( Q' \). The definition of \( T_A \) implies that \( Q'' \models \delta(s, \sigma) \). Lemma 10 implies that \( S \setminus Q'' \neq \delta(s, \sigma) \). Thus, \( Q'' \notin \delta_T(Q', \sigma) \), contradiction. \( \square \)

B Proofs for Section 4

Lemma 6 \( P_M(L(Q)) = \sum_{x \in X} P_I(x) \cdot P(x, Q) \).

Proof. We partition the language \( L(Q) \) according to the first letter of the word. Then, we have that \( P_M(L(Q) \cap \{ w \mid w \text{ starts with } x \}) = P_I(x) \cdot P(x, Q) \). Thus, \( P_M(L(Q)) = \sum_{x \in X} P_I(x) \cdot P(x, Q) \). \( \square \)

Let \( x \) be a state of a Markov chain with \( P_I(x) > 0 \). Then for every state \( (x, Q) \) we have that if \( (x, Q) \) is probable, then \( P_M(L(Q)) > 0 \). Thus we conclude the following lemma.
Lemma 21. Let \( s \) be a state of an AWW \( \mathcal{A} \) and let \( \mathcal{M} \) be a Markov chain. Then, \( P_M(L(s)) < 1 \) iff there exists a state \( x \) of \( \mathcal{M} \) and a set \( Q \subseteq S \) s.t. \( P_I(x) > 0 \), \( s \not\in Q \) and the state \( (x, Q) \) is probable.

Proof. Let \( w \) be a word in \( \overline{L(s)} \), then \( s \not\in \text{type}_A(w) \). By Lemma 20, \( w \in L(Q) \) for some \( Q \) s.t. \( s \not\in Q \). This implies that \( L(s) \subseteq \cup_{s \in Q} L(Q) \). By Theorem 2, for every \( Q \) s.t. \( s \not\in Q \) we have \( L(Q) \subseteq \overline{L(s)} \), thus, \( \overline{L(s)} = \cup_{s \in Q} L(Q) \). Then, \( P_M(L(s)) < 1 \) iff \( P_M(\overline{L(s)}) > 0 \) iff there exists a set \( Q \) s.t. \( s \not\in Q \) and \( P_M(L(Q)) > 0 \) iff there exists \( Q \) s.t. \( s \not\in Q \), a state \( x \) such that \( P_I(x) > 0 \), and \( P(x, Q) > 0 \). \( \square \)

B.1 Proof of Theorem 3

Theorem 3 Let \( C \) be an SCC of \( G \). Then \( C \) is a BSCC of \( H \) iff it satisfies the following conditions:

1. \( C \) corresponds to a BSCC \( K \) of \( \mathcal{M} \).
2. Every finite path of \( K \) is a projection of a path in \( C \).
3. \( C \) contains a fulfilling path.

In the rest of the section we prove theorem 3. We start by showing that every BSCC of \( H \) satisfies the conditions of the theorem. Lemma 21 makes a connection between the paths of \( \mathcal{M} \) and the paths of \( H \).

Lemma 21. Let \( C \) be a BSCC of \( H \). Let \( (x, Q) \) be a state in \( C \), and let \( w \) be a path of \( \mathcal{M} \) that starts at \( x \) and has type \( Q \). Then, with probability one, the augmented path of \( w \) is contained in \( C \).

Proof. Let \( \Theta = \{ w | w \) starts at \( x \), has type \( Q \), and its augmented path is not contained in \( C \} \). We prove that \( P_{M(w)}(\Theta) = 0 \). First, note that all the augmented paths of the paths in \( \Theta \) start at \( (x, Q) \). Since non-probable states do not have probable successors, for every word \( w \) in \( \Theta \) there exists a unique \( i \) such that the \( i \)th state in the augmented path of \( w \) is in \( H \) and the \( i + 1 \)th state of the augmented path of \( w \) is not. We define

\[ \Theta_i = \{ w | w \) starts at \( x \), has type \( Q \), and its augmented path exits \( C \) in its \( i \)th state \}, \]

then we have \( \Theta = \cup_{i=1}^\infty \Theta_i \). Thus, it is sufficient to prove that for every \( i \) we have \( P_{M(w)}(\Theta_i) = 0 \). We further partition \( \Theta_i \) according to the type \( Q' \) of the suffix of \( w \) that starts at \( w_{i+1} \) and the prefix \( \rho = w_0, w_1, \ldots, w_{i+1} \). Note that \( (w_{i+1}, Q') \) is not probable. For a set \( Q' \) of states and a finite path \( \rho \) of length \( i + 1 \), we define

\[ \Theta_i^{Q', \rho} = \{ w | Q' \) is the type of \( w_{i+1} \), \( \rho = w_0, w_1, \ldots, w_{i+1} \), and \( P(w_{i+1}, Q') = 0 \} \]

Then \( \Theta_i \subseteq \cup_{\rho \in X_{i+1}, Q' \subseteq S} \Theta_i^{Q', \rho} \). For the set \( \Theta_i^{Q', \rho} \) we have \( P_{M(w)}(\Theta_i^{Q', \rho}) \leq P_{M(w)}(\rho) \cdot P(w_{i+1}, Q') \). Since \( P(w_{i+1}, Q') = 0 \), we have that \( P_{M(w)}(\Theta_i^{Q', \rho}) = 0 \). This implies that \( P_{M(w)}(\Theta_i) = 0 \), so \( P_{M(w)}(\Theta) = 0 \). \( \square \)

Lemma 22. Let \( C \) be a BSCC of \( H \) that corresponds to an SCC \( K \) of \( \mathcal{M} \). Then \( K \) is a BSCC of \( \mathcal{M} \).
Proof. Let \((x, Q)\) be a state in a BSCC \(C\) of \(H\). Suppose \(K\) is not a BSSC of \(M\). Then a path \(\rho\) starting from \(x \in K\) almost surely leaves \(K\), meaning it has a suffix with no letters in \(K\). Since \((x, Q)\) is in \(H\), with positive probability \(\rho\) has type \(Q\). This implies that with positive probability the augmented path of \(\rho\) starts at \((x, Q)\) and its projection has a suffix with no letters in \(K\). This implies with positive probability the augmented path of \(\rho\) starts at \((x, Q)\) and exits \(C\), meaning exits \(H\). This contradicts Lemma 21. \(\boxempty\)

Lemma 23. Let \(C\) be a BSCC of \(H\) that corresponds to a BSCC \(K\) of \(M\). Then every finite path in \(K\) is the projection of some path in \(C\).

Proof. Suppose that \(x\) is a finite path in \(K\) that is not the projection of a path in \(C\). Let \(x\) be a state in \(K\), and let \((x, Q)\) be a state in \(C\). Since \((x, Q)\) is in \(C\), with positive probability a path \(w\) of \(M\) that starts at \(x\) has type \(Q\). By ergodicity, a path that starts at \(x\) almost surely eventually takes \(\alpha\). Thus, with positive probability, a path \(w\) that starts at \(x\) has type \(Q\) and contains \(\alpha\). Let \(\rho\) be the augmented path of a path \(w\) that starts at \(x\) and has type \(Q\). By the definition of augmented path, \(\rho\) starts at \((x, Q)\), and since the projection of \(\rho\) contains \(\alpha\), \(\rho\) exits \(C\). This implies that with positive probability the augmented path of a path of \(M\) that starts at \(x\) exits \(C\), which contradicts Lemma 21. \(\boxempty\)

We now prove that every BSCC of \(H\) contains a fulfilling path. We start by showing that every augmented path in \(G\) contains a fulfilling path.

Lemma 24. Let \(w\) be a path in \(M\). Then the augmented path \(\rho_G\) of \(w\) contains a fulfilling path.

Proof. Let \(Q_1\) be the type of \(w\). By Lemma 20, there exists an accepting run \(\rho_f\) of \(\mathcal{A}(Q_1, Q_2, \emptyset, \emptyset)\) on \(w\). For every \(i\), since \(\rho_f^i\) is a run of \(\mathcal{A}(Q_1, Q_2, Q_3, Q_4)\) on \(w^i\), the first element \(Q_1'\) of the \(i\)th state of \(\rho_f^i\) is \(\text{type}_\mathcal{A}(w^i)\). Together with the definition of augmented path this implies that \(\pi_1(\rho_f^i) = \pi_3(\rho_G)\). Since \(\rho_f^i\) is an accepting run, it contains infinitely many accepting states of the form \((Y_1, Y_2, \emptyset, \emptyset)\). Recall that from a state of the form \((Y_1, Y_2, \emptyset, \emptyset)\) all the outgoing transitions in \(\mathcal{A}_f\) enter states of the form \((Y_1', Y_2', (Y_1' \setminus F), (Y_2' \setminus F))\). This implies that \(\rho_G\) contains a fulfilling path. \(\boxempty\)

Lemma 25. Let \(C\) be a BSCC of \(H\), then \(C\) contains a fulfilling path.

Proof. Let \((x, Q)\) be a state in \(C\). Since \(C\) is in \(H\), \((x, Q)\) is probable. Thus, with positive probability, a path that starts with \(x\) is of type \(Q\). Lemma 21 implies that with probability one an augmented path of a path that starts in \(x\) and has a type \(Q\) is contained in \(C\). Thus, with positive probability, the augmented path of a path that starts at \(x\) is contained in \(C\). Since there is at least one outgoing path out of \(x\), this implies that there exists a path that starts at \(x\) such that its augmented path is contained in \(C\). Together with Lemma 24, this implies that \(C\) contains a fulfilling path. \(\boxempty\)

This completes the proof that of the first direction of Theorem 3. In the rest of this section we prove the other direction of Theorem 3. Let \(C\) be an SCC of \(G\), we prove that if \(C\) satisfies the conditions of the theorem, then \(C\) is a BSCC of \(H\). Let \(C\) be an SCC of \(G\) that corresponds to a BSCC \(K\) of \(M\) such that every finite path in \(K\) is a projection of a path in \(C\). The next four lemmas prove that if \(C\) contains a fulfilling path, then \(C\) is a BSCC of \(H\).
Lemma 26. Let $C$ be an SCC of $G$ that corresponds to a BSCC $K$ of $M$ such that every finite path of $K$ is a projection of a path in $C$. Let $\rho_C$ be an infinite path in $C$ and let $\rho_K$ be the projection of $\rho_C$. If $\rho_K$ contains every finite path in $K$ infinitely many times, then $\rho_C$ contains every finite path in $C$ infinitely many times.

Proof. Every suffix of $\rho_K$ contains every finite path in $K$ infinitely many times. Assume that $\rho_C$ contains some finite path $\alpha$ of $C$ only finitely many times, then there exists a suffix $\rho'_K$ of $\rho_C$ such that its projection $\rho'_K$ on $M$ contains every finite path in $K$ infinitely often and $\rho'_K$ does not contain $\alpha$ at all. Thus, it is enough to prove that $\rho_C$ contains every finite path of $C$. Assume now to the contrary that there is a finite path $\alpha$ in $C$ that is not contained in $\rho_C$. For every finite path $\rho$ in $C$ we define $\Xi(\rho)$ to be the set of finite paths $\rho'$ in $C$ that have the same projection on $K$ as $\rho$ and do not contain $\alpha$. Let $\xi(\rho)$ be the set of states in $C$ that are the last states of the paths in $\Xi(\rho)$.

We define a sequence $\alpha_0, \alpha_1, \ldots$ of paths in $C$ such that for every $i$ such that $|\xi(\alpha_i)| \geq 1$, we have that $|\xi(\alpha_i)| > |\xi(\alpha_{i+1})|$, and $\alpha_i$ contains $\alpha$. Since $\xi(\alpha_0)$ is a finite set, there exists $n$ such that $|\xi(\alpha_n)| = 0$. Let $\beta_i$ be the projection of $\alpha_i$, for $i \geq 0$. Then all paths in $C$ with projection $\beta_n$ contain $\alpha$. Since $\rho_K$ contains $\beta_n$, we have that $\rho_C$ contains $\alpha$, contradiction.

We define $\alpha_0 = \alpha$. Let $\alpha_i$ be the $i$'th path in the sequence. Let $(x, Q)$ be a state in $\xi(\alpha_i)$, and let $\rho$ be the path in $\Xi(\alpha_i)$ (so it has projection $\beta_i$) that ends in $(x, Q)$. We define $\alpha_{i+1} = \alpha_\gamma \rho$ where $\gamma$ is some path that connect $\alpha$ to $\rho$. Since $C$ is an SCC, there exists such a path. Note that $\alpha_{i+1}$ has a suffix with projection $\beta_i$ thus $\xi(\alpha_{i+1}) \subseteq \xi(\alpha_i)$, furthermore, the path with projection $\beta_{i+1}$ that ends in $(x, Q)$ contains $\alpha$, thus $(x, Q) \notin \xi(\alpha_{i+1})$. $\square$

Lemma 19, Lemma 4, and the definition of $G$ imply the following lemma.

Lemma 27. Let $\rho_G = (x^0, Q^0), (x^1, Q^1), \ldots$ be a finite or infinite path in $G$, then:

1. Let $\rho_f$ be a run of $A_f$ such that $\pi_1(\rho_f) = \pi_2(\rho_g)$. Then $\rho_f$ is a run of $A_f$ on $\pi_1(\rho_G)$.

2. The sequence $(Q^0, S \setminus Q^0, Q^0 \setminus F, (S \setminus Q^0) \setminus F, (Q^1, S \setminus Q^1, Q^1 \setminus F, (S \setminus Q^1) \setminus F), (Q^2, S \setminus Q^2, Q^2 \setminus F, (S \setminus Q^2) \setminus F), \ldots$ is a run of $A_f$ on $x^0, x^1, \ldots$.

Lemma 28. Let $\alpha$ be a fulfilling path in $G$. Let $\rho_G$ be a path in $G$ that starts at a state $(x, Q_1)$ and contains $\alpha$ infinitely often. Then $A_f^{(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus F))}$ accepts the projection $\pi_1(\rho_G)$ of $\rho_G$ on $M$.

Proof. We show an accepting run of $A_f^{(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus F))}$ on $\pi_1(\rho_G)$. Let $i_1 = 0$ and $i_2, i_3, \ldots$ be an infinite sequence of indices such that for every $j \geq 1$, we have that the subpath $\rho_G^{i_2j}, \rho_G^{i_2j+1}, \ldots, \rho_G^{i_2j+1}$ of $\rho_G$ is $\alpha$. Let $\alpha_f$ be the path of $A_f$ that corresponds to $\alpha$ as defined in Definition 14.

We define a run of $A_f$ as follows, for every $j \geq 1$, the part of the run that starts at $i_2j$ and ends at $i_2j+1$ is $\alpha_f$. For every $j \geq 0$ and $i_2j+1 < k < i_2j+2$, the $k$'th state of the run is $Q^k_{i_2j} = \rho_G^{i_2j}, Q^k_{i_2j+1} = S \setminus Q^k_{i_2j}, Q^k_{i_2j+2} = Q^k_{i_2j+1} \setminus F$, and $Q^k_{i_2j+2} = Q^k_{i_2j+1} \setminus F$.

Lemma 27 implies that the path that we describe is a run of $A_f^{(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus F))}$ on the projection of $\rho_G$. Since $\rho_G$ contains infinitely many fulfilling paths, the run contains infinitely many accepting states. $\square$
Lemma 29. Let \( C \) be an SCC of \( G \) that corresponds to a BS\( \overline{C} \)C \( K \) of \( \mathcal{M} \) and satisfies the conditions below.

- Every finite path of \( K \) is a projection of a path in \( C \).
- \( C \) contains a fulfilling path.

Then, \( C \) is a BS\( \overline{C} \)C \( H \).

Proof. First we prove that \( C \) is contained in \( H \). Since for every SCC \( C \) of \( G \) either \( C \subseteq H \) or \( C \cap H = \emptyset \), it is enough to prove that \( C \) contains at least one probable state. We prove that with probability one a path \( \rho_K \) that starts from a state \( x \) in \( K \) is a projection of a path \( \rho_C \) that starts at a state \( (x, Q_1) \) in \( C \), such that \( A_f^{(Q_1, S \setminus Q_1, (Q_1 \setminus F), (S \setminus Q_1 \setminus F))} \) accepts \( \rho_K \), that is, \( t_{v, v}(\rho_K) = Q_1 \). This implies that with probability one, a path that starts from a state \( x \) in \( K \) has type \( Q_1 \) such that \( (x, Q_1) \) is in \( C \). This implies that at least one state in \( C \) is probable.

Let \( \rho_K \) be a path of \( K \) that starts at \( x \). Let \( \rho_C \) be a path in \( C \) with projection \( \rho_K \) on \( K \). Then, by ergodicity (see Section 2.2) \( \rho_K \) contains every finite path in \( K \) infinitely often. Lemma 26 implies that \( \rho_C \) contains every finite path in \( C \) infinitely often, and in particular it contains the fulfilling path infinitely often. Lemma 28 implies that \( A_f^{(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus F))} \) accepts \( \rho_K \).

Finally, we prove that \( C \) is a BS\( \overline{C} \)C \( H \). Since \( C \) corresponds to a BS\( \overline{C} \)C \( K \) of \( \mathcal{M} \), every successor SCC \( C' \) of \( C \) in \( G \) corresponds to \( K \) too. Lemma 8 implies that there exists a finite path in \( K \) that is not the projection of a path in \( C' \). By Lemma 23 \( C' \) is not a BS\( \overline{C} \)C \( H \). Since \( C \) is a BS\( \overline{C} \)C \( H \) and no successor SCC of \( C \) is a BS\( \overline{C} \)C \( H \), we conclude that \( C \) is a BS\( \overline{C} \)C of \( H \). ☐

C. Proofs for Section 5

Lemma 30. Let \( \rho \) be a path in \( G_f \), then the projection of the states of \( \rho \) on \( G \) is a path in \( G \).

Proof. Let \( ((x, (Q_1, Q_2, Q_3, Q_4)), (x', (Q_1', Q_2', Q_3', Q_4'))) \) be a transition in \( G_f \), we prove that \( ((x, Q_1), (x', Q_1')) \) is a transition of \( G \). The definition of \( G_f \) implies that \( \rho_T(x, x') > 0 \) and that \( (Q_1, Q_2, Q_3, Q_4') \in \delta_f((Q_1, Q_2, Q_3, Q_4), x) \). Lemma 4 implies that \( (Q_1, x, Q_1') \) is a transition in \( \delta_A \). By the definition of \( G \), we have that \( ((x, Q_1), (x', Q_1')) \) is a transition of \( G \). ☐

Lemma 9 An SCC \( C \) of \( G \) contains a fulfilling path iff there exists a path \( (x_0, (Q_0^1, Q_0^2, Q_0^3, Q_0^4)), (x_1, (Q_1^1, Q_1^2, Q_1^3, Q_1^4)), \ldots, (x_n, (Q_n^1, Q_n^2, Q_n^3, Q_n^4)) \) in \( G_f \) such that the path \( (x_0, Q_0^1), (x_1, Q_1^1), \ldots, (x_n, Q_n^1) \) is contained in \( C \), \( Q_3^0 = Q_1^0 \setminus F \), and \( Q_4^0 = Q_2^0 \setminus F \).

Lemma 9 is followed directly from Lemma 30.

Example 6. We present in Figure 4 the product \( G_{2f} \) of \( A_f \) and \( M_2 \) which are presented in Figures 1 and 2. It easy to see that the only SCC in \( G_2 \) that contains a fulfilling path is the SCC of \( (a, \{1\}) \) and \( (b, \emptyset) \) that contains the path \( (a, \{1\}), (b, \emptyset) \) that correspond to the path \( (a, \{1\}), (\{1\}, \emptyset) \) of \( G_{2f} \). ☐
D Exact Probability of $L(A)$

In order to compute the exact probability $P_M(L(A))$, we prove that the graph $H$ can be defined as a Markov chain that refines $M$. First we define a Markov chain over $H$. Define the initial probability $P_H(I(x,Q))$ as $P_I(x) \cdot P(x,Q)$. Define the probability of a transition $(x,Q) \rightarrow (x',Q')$ to be

$$P_T((x,Q),(x',Q')) = \frac{P_T(x,x')P(x',Q')}{P(x,Q)}.$$

**Lemma 31.** $H$ is a Markov Chain.

**Proof.** The sum of the initial probabilities in $H$ is equal to

$$\sum_{(x,Q) \in G} P_I(x)P(x,Q) = \sum_{x} P_I(x) \sum_{Q} P(x,Q)$$

Since every path that starts at $x$ has a unique type, we have $\sum_{Q} P(x,Q) = 1$. Thus,

$$\sum_{x} P_I(x) \sum_{Q} P(x,Q) = \sum_{x} P_I(x) = 1.$$

Next, we show that for every state in $H$, the probabilities of its outgoing transitions are sum to one. Consider $(x,Q) \in H$. Then, the sum of the probabilities of the outgoing transitions from $(x,Q)$ is:

$$\sum_{(x,Q) \rightarrow (x',Q')} \frac{P_T(x,x')P(x',Q')}{P(x,Q)} = \frac{1}{P(x,Q)} \cdot \sum_{x \rightarrow x'} P_T(x,x') \cdot \sum_{Q \rightarrow Q'} P(x',Q')$$
Thus, it is enough to prove that
\[ \sum_{x \rightarrow x'} P_T(x, x') \cdot \sum_{Q \rightarrow Q'} P(x', Q') = P(x, Q). \]

We can partition the probability that a path that starts at \( x \) is accepted by \( Q \) according to the second letter of the path. For every \( x' \) the probability that a path starts at \( x \) its second letter is \( x' \) and it is accepted by \( Q \) is \( P_T(x, x') \cdot \sum_{Q' \rightarrow Q} P(x', Q') \). 

Next, we prove that \( H \) refines \( \mathcal{M} \). We start by proving that for every regular language \( L \) we have \( P_M(L) = P_H(L) \). We define a labelling function \( V \) for \( H \) such that \( V(x, Q) = x \) thus both Markov chains are defined over the same alphabet.

**Lemma 32.** For every \( \omega \)-regular language \( L \) we have \( P_M(L) = P_H(L) \).

**Proof.** It is sufficient to show that \( \mathcal{M} \) and \( H \) agree on the basic cylindrical sets. That is, for every finite word \( w = x_0, x_1, \ldots, x_k \), we have \( P_M(\Delta(w)) = P_H(\Delta(w)) \). Recall that \( T_A \) is reverse deterministic. So for a fixed \( w = x_0, \ldots, x_k \), there exists a unique path \( \rho = (x_0, Q_0), \ldots, (x_k, Q_k) \) such that \( V(\rho) = w \) and \( \rho \) ends in \( (x_k, Q_k) \). We define \( \rho^w.Q \) to be the unique path in \( H \) such that \( V(\rho^w.Q) = w \) and \( \rho_k^w.Q = Q \). So,

\[
P_H(\Delta(w)) = \sum_{V(\rho) = w} P_H(\rho), \text{ we partition according to the } k \text{th state of } \rho
\]

\[
= \sum_Q P_H(\rho_w.Q)\prod_{0 \leq i < k} P_H(\rho_i^w.Q, \rho_{i+1}^w.Q)
\]

\[
= \sum_Q \left[ P_L(x_0) P(\rho_0^w.Q) \prod_{0 \leq i < q} P_T(x_i, x_{i+1}) \frac{P(\rho_{i+1}^w.Q)}{P(\rho_i^w.Q)} \right], \text{ definition of } P_T^H
\]

\[
= \sum_Q \left[ P_L(x_0) \prod_{0 \leq i < k} P_T(x_i, x_{i+1}) \right] P(\rho_k^w.Q)
\]

\[
= \sum_Q P_M(w) \cdot P(\rho_k^w.Q) \text{ since } \rho_k^w.Q = (x_k, Q),
\]

\[
= P_M(w) \square
\]

In order to complete the proof that \( H \) refines \( \mathcal{M} \) we need to prove that 

\[
P_M(L(Q_1, Q_2, Q_3, Q_4)) = \sum_{x \in X} P_H^I(x, Q_1).
\]

This is proved in Lemma 33.

**Lemma 33.** \( P_M(L((Q_1, Q_2, Q_3, Q_4))) = \sum_{x \in X} P_H^I(x, Q_1). \)

**Proof.** Lemma 32 implies that \( P_M(\{w|w_0 = x \text{ and } w \text{ has type } Q_1\}) = P_H(\{w|w_0 = x \text{ and } w \text{ has type } Q_1\}) \). We need to prove that \( P_H(\{w|w_0 = x \text{ and } w \text{ has type } Q_1\}) = P_H^I(x, Q_1) \). It is enough to prove that a projection of a path that starts in \( (x, Q_1) \) is with...
probability one of type \(Q_1\) meaning is accepted with probability one by 
\(A_f^{(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus F))}\). Let \(\rho\) be a path that starts at \((x, Q_1)\). By ergodicity, \(\rho\) reaches 
a BSCC of \(H\) and then pass through every finite path of the BSCC infinitely often. 
In particular, \(\rho\) pass infinitely often through a fulfilling path. Lemma 28 implies that 
\(A_f^{(Q_1, Q_2, (Q_1 \setminus F), (Q_2 \setminus F))}\) accepts the projection of \(\rho\) on \(M\) \(\square\)

Given a state \(s\) of \(A\), we have \(L(s) = \cup_{s \in Q} L(Q)\), which implies that 
\(P_M(L(s)) = \sum_{s \in Q} \sum_{x \in X} P^H(x, Q)\). Thus, it left to calculate the exact probabilities of \(H\). We need to compute for every state \((x, Q)\) of \(H\), the probability \(P(x, Q)\). In [10] it is shown that 
these probabilities form a unique solution to the following linear system:

1. for every state \((x, Q)\) of \(H\) we have \(P(x, Q) = \sum_{(x, Q), (x', Q')} \in G \ P_T(x, x') P(x', Q')\).
2. For every state \(x\) of \(M\) we have \(\sum_{Q \in 2^S} P(x, Q) = 1\).