A Saturation Method for the Modal Mu-Calculus with Backwards Modalities over Pushdown Systems

M. Hague[a], C.-H. L. Ong[a]

[a] *Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD, UK*

Abstract

We present an algorithm for computing *directly* the denotation of a modal μ-calculus formula χ with backwards modalities over the configuration graph of a pushdown system. Our method gives the first extension of the saturation technique to the full modal μ-calculus with backwards modalities. Finite word automata are used to represent sets of pushdown configurations. Starting from an initial automaton, we perform a series of automaton manipulations which compute the denotation by recursion over the structure of the formula. We introduce notions of under-approximation (soundness) and over-approximation (completeness) that apply to automaton transitions rather than runs. Our algorithm is relatively simple and direct, and avoids an immediate exponential blow up.

Keywords: Modal Mu-Calculus, Backwards Time, Pushdown Systems, Parity Games, Winning Regions. Global Model Checking. Saturation Methods.

1. Preliminaries

1.1. Pushdown Systems

A ***pushdown system*** (PDS) is a triple $\mathbb{P} = (\mathcal{P}, \mathcal{D}, \Sigma_\perp)$ where $\mathcal{P}$ is a set of control states, $\Sigma_\perp := \Sigma \cup \{\perp\}$ is a finite stack alphabet (we assume $\perp \notin \Sigma$), $\mathcal{D} \subseteq \mathcal{P} \times \Sigma_\perp \times \mathcal{P} \times \Sigma_\perp^*$ is a set of pushdown rules. As is standard, we assume that the bottom-of-stack symbol $\perp$ is neither pushed onto, nor popped from, the stack. We write $\langle p, aw \rangle \hookrightarrow \langle p', w'w \rangle$ whenever $p\, a \rightarrow p'\, w' \in \mathcal{D}$ and $\mathcal{C}$ to refer to the set of all pushdown configurations.

1.2. Modal μ-Calculus

Given a set of propositions AP and a disjoint set of variables $\mathcal{Z}$, formulas of the modal μ-calculus are defined as follows (with $x \in AP$ and $Z \in \mathcal{Z}$):

$$\varphi \ := \ x \mid \neg x \mid Z \mid \varphi \ \wedge \ \varphi \mid \varphi \ \vee \ \varphi \mid \Box\varphi \mid \Diamond\varphi \mid \mu Z.\varphi \mid \nu Z.\varphi \ .$$

Thus we assume that the formulas are in *positive form*, in the sense that negation is only applied to atomic propositions. Over a pushdown system, the semantics of a formula φ are given with respect to a *valuation* $V : \mathcal{Z} \rightarrow \mathcal{P}(\mathcal{C})$ which maps each free variable to its set of satisfying configurations and an environment $\rho : AP \to \mathcal{P}(\mathcal{C})$ mapping each atomic proposition to its set of satisfying configurations. We then have,

$$
\begin{array}{rcl}
[\![x]\!]^{\mathbb{P}}_V & = & \rho(x) \\
[\![\neg x]\!]^{\mathbb{P}}_V & = & \mathcal{C} \setminus \rho(x) \\
[\![Z]\!]^{\mathbb{P}}_V & = & V(Z) \\
[\![\varphi_1 \wedge \varphi_2]\!]^{\mathbb{P}}_V & = & [\![\varphi_1]\!]^{\mathbb{P}}_V \cap [\![\varphi_2]\!]^{\mathbb{P}}_V \\
[\![\varphi_1 \vee \varphi_2]\!]^{\mathbb{P}}_V & = & [\![\varphi_1]\!]^{\mathbb{P}}_V \cup [\![\varphi_2]\!]^{\mathbb{P}}_V \\
[\![\Box\varphi]\!]^{\mathbb{P}}_V & = & \left\{ c \in \mathcal{C} \;\middle|\; \forall c'.c \hookrightarrow c' \Rightarrow c' \in [\![\varphi]\!]^{\mathbb{P}}_V \right\} \\
[\![\Diamond\varphi]\!]^{\mathbb{P}}_V & = & \left\{ c \in \mathcal{C} \;\middle|\; \exists c'.c \hookrightarrow c' \wedge c' \in [\![\varphi]\!]^{\mathbb{P}}_V \right\} \\
[\![\overline{\Box}\varphi]\!]^{\mathbb{P}}_V & = & \left\{ c \in \mathcal{C} \;\middle|\; \forall c'.c' \hookrightarrow c \Rightarrow c' \in [\![\varphi]\!]^{\mathbb{P}}_V \right\} \\
[\![\overline{\Diamond}\varphi]\!]^{\mathbb{P}}_V & = & \left\{ c \in \mathcal{C} \;\middle|\; \exists c'.c' \hookrightarrow c \wedge c' \in [\![\varphi]\!]^{\mathbb{P}}_V \right\} \\
[\![\mu Z.\varphi]\!]^{\mathbb{P}}_V & = & \bigcap \left\{ S \subseteq \mathcal{C} \;\middle|\; [\![\varphi]\!]^{\mathbb{P}}_{V[Z\mapsto S]} \subseteq S \right\} \\
[\![\nu Z.\varphi]\!]^{\mathbb{P}}_V & = & \bigcup \left\{ S \subseteq \mathcal{C} \;\middle|\; S \subseteq [\![\varphi]\!]^{\mathbb{P}}_{V[Z\mapsto S]} \right\}
\end{array}
$$

where $V[Z \mapsto S]$ updates the valuation V to map the variable Z to the set S.

The operators $\Box\varphi$ and $\Diamond\varphi$ assert that φ holds after all possible transitions and after some transition respectively; $\overline{\Box}$ and $\overline{\Diamond}$ are their backwards time counterparts; and the μ and ν operators specify greatest and least fixed points. Another interpretation of these operators is given below. For a full discussion of the modal μ-calculus we refer the reader to a survey by Bradfield and Stirling [7].

1.3. Approximants

Thanks to the Knaster-Tarski Fixed Point Theorem, the semantics of a fixed point formula $[\![\sigma Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V$ where $\overline{Y} = Y_1, \ldots, Y_n$ and $\sigma \in \{\mu, \nu\}$ can be given as the limit of the sequence of α-***approximants*** $[\![\sigma^\alpha Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V$, where α ranges over the ordinals and λ ranges over the limit ordinals:

$$
\begin{array}{rcl}
[\![\sigma^0 Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V & := & \mathit{Init} \\
[\![\sigma^{\alpha+1} Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V & := & [\![\chi(\overline{Y}, Z)]\!]^{\mathcal{G}}_{V[Z\mapsto[\![\sigma^\alpha Z.\chi(\overline{Y},Z)]\!]^{\mathbb{P}}V]} \\
[\![\sigma^\lambda Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V & := & \bigcirc_{\alpha<\lambda}[\![\sigma^\alpha Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V
\end{array}
$$

where $\mathit{Init} = \emptyset$ and $\bigcirc = \bigcup$ when $\sigma = \mu$, and Init is the set of all configurations and $\bigcirc = \bigcap$ when $\sigma = \nu$. The least ordinal κ such that $[\![\sigma^\kappa Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V = [\![\sigma Z.\chi(\overline{Y}, Z)]\!]^{\mathbb{P}}_V$ is called the *closure ordinal*.

Example 1.1. When interpreted in a pushdown graph, $[\![\sigma^\alpha Z.\chi(\overline{Y}, Z)]\!]_{\alpha\in\mathbf{Ord}}$ may have an infinite closure ordinal. Consider the pushdown graph in Figure 1 (which is a dual of an example of Cachat's [20]). The proposition p is true only when the control state is p and f is true only at control state f. In this graph $[\![\mu Z_1.\nu Z_2.\,(p \wedge \Box Z_1) \vee (f \wedge \Box Z_2)]\!]$ consists of all configurations. However, any $\langle f, a\, a^n\bot\rangle$ for some n only appears in an approximant of the least fixed point when $\langle f, a\, a\, a^n\bot\rangle$ and $\langle p, a\, a^n\bot\rangle$ appear in the previous approximant (since $\Box Z_2$ quantifies over all transitions from $\langle f, a\, a^n\bot\rangle$). Hence, all $\langle p, a^n\bot\rangle$ must appear

$$\begin{tikzcd}
\langle f, \bot \rangle \arrow[loop left] & \langle f, a\bot \rangle \arrow[r] \arrow[d] & \langle f, aa\bot \rangle \arrow[r] \arrow[d] & \cdots \\
\langle p, \bot \rangle \arrow[u] & \langle p, a\bot \rangle \arrow[l] & \langle p, aa\bot \rangle \arrow[l] & \cdots \arrow[l]
\end{tikzcd}$$

Figure 1: The configuration graph of an example pushdown system.

in the α-approximant before any $\langle f, a^n \bot \rangle$ can appear in the $(\alpha+1)$-approximant. Thus the first approximant containing all p configurations is the ω-approximant. It follows that the least fixed point in question has an infinite closure ordinal. Cachat also shows that a greatest fixed point may also have an infinite closure ordinal.

1.4. Alternating Multi-Automata

We use alternating multi-automata [2] as a representation of (regular) sets of configurations. Given a pushdown system $(\mathcal{P}, \mathcal{D}, \Sigma)$ with $\mathcal{P} = \{p^1, \ldots, p^z\}$, an ***alternating multi-automaton*** A is a quintuple $(\mathcal{Q}, \Sigma, \Delta, I, \mathcal{F})$ where $\mathcal{Q}$ is a finite set of states, $\Delta \subseteq \mathcal{Q} \times (\Sigma \cup \{\bot\}) \times 2^{\mathcal{Q}}$ is a set of transitions (we assume $\bot \notin \Sigma$), $I = \{q^1, \ldots, q^z\} \subseteq \mathcal{Q}$ is a set of initial states, and $\mathcal{F} \subseteq \mathcal{Q}$ is a set of final states. Observe that there is an initial state for each control state of the pushdown system. We write $q \xrightarrow{a} Q$ just if $(q, a, Q) \in \Delta$; and define $q \xrightarrow{\varepsilon} \{q\}$; and $q \xrightarrow{aw} Q_1 \cup \cdots \cup Q_n$ just if $q \xrightarrow{a} \{q_1, \ldots, q_n\}$ and $q_k \xrightarrow{w} Q_k$ for all $1 \leq k \leq n$. Finally we define the *language accepted by* A, $\mathcal{L}(A)$, by: $\langle p^j, w \rangle \in \mathcal{L}(A)$ just if $q^j \xrightarrow{w} Q$ for some $Q \subseteq \mathcal{F}$. We further define $\mathcal{L}_q(A)$ to be the set of all words accepted from the state q in A. Henceforth, we shall refer to alternating multi-automata simply as ***automata***. In cases of ambiguity, we may specify runs of a particular automaton A with a transition relation Δ by $q \xrightarrow[A]{a} Q$ and $q \xrightarrow[\Delta]{a} Q$ respectively.

1.5. Reachability and Projection

Formulas of the form $\Box\varphi$ and $\Diamond\varphi$ assert a one-step backwards reachability property, which we compute using a simplification of the reachability algorithm [2] due to Bouajjani *et al.*. Cachat's extension of this algorithm to Büchi games [19] requires a technique called *projection*. Using an example, we briefly introduce the relevant techniques.

Take a PDS with the rules $p^1\, a \rightarrow p^2\, \varepsilon$ and $p^2\, b \rightarrow p^2\, ba$. The automaton A_{eg} in Figure 2 (with q_f being the only accepting state) represents a configuration set $\mathcal{C}$. Let $Pre(\mathcal{C})$ be the set of all configurations that can reach $\mathcal{C}$ in exactly one step. To calculate $Pre(\mathcal{C})$ we first add a new set of initial states — since we don't necessarily have $\mathcal{C} \subseteq Pre(\mathcal{C})$. By applying $p^1\, a \rightarrow p^2\, \varepsilon$, any configuration of the form $\langle p^1, aw \rangle$, where w is accepted from q^2 in A_{eg}, can reach $\mathcal{C}$. Hence we add an a-transition from q^1_{new}. (Via the pop transition, we reach $\langle p^2, w \rangle \in \mathcal{L}(A_{eg})$.) Alternatively, via $p^2\, b \rightarrow p^2\, ba$, any configuration of the form $\langle p^2, bw \rangle$, where baw

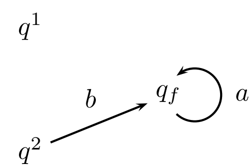

Figure 2: The automaton A_{eg} accepting $\langle p^2, ba^* \rangle$.

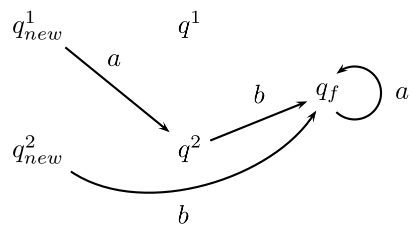

Figure 3: A_{eg} updated by the rules $p^1\, a \rightarrow p^2\, \varepsilon$ and $p^2\, b \rightarrow p^2\, ba$.

is accepted from q^2 in A_{eg}, can reach $\mathcal{C}$. The push, when applied backwards, replaces ba by b. We add a b-transition from q^2_{new} which skips any run over ba from q^2. Figure 3 shows the resulting automaton.

To ensure termination of the Büchi construction, Cachat uses *projection*, which replaces a new transition to an old initial state with a transition to the corresponding new state. Hence, the transition in Figure 3 from q^1_{new} is *replaced* by the transition in Figure 4. The old initial states are then unreachable, and deleted, which, in this case, leaves an automaton with the same states as Figure 2 (modulo the *new* suffix) but an additional transition. In this sense, the state-set remains fixed.

2. The Algorithm

Without loss of generality, assume all pushdown commands are $p\, a \rightarrow p'\, \varepsilon$, $p\, a \rightarrow p'\, b$, or $p\, a \rightarrow p'\, bb'$.

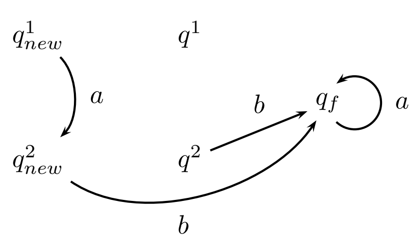

Figure 4: The result of projecting the automaton in Figure 3.

We begin by introducing some notation. A literal $\widehat{x}$ is either x or $\neg x$ for an atomic proposition x. For a modal μ-calculus formula χ, we write $FV(\chi)$ for the set of free variables of χ. Henceforth we fix a modal μ-calculus formula χ. We shall assume χ contains no sub-formulas of the form $\sigma Z.\widehat{x}$ or $\sigma Z.X$ with $\sigma \in \{\mu, \nu\}$. Furthermore, all bound variable names are unique.

The algorithm is given in Procedures 1 to 11. Each procedure returns an automaton and a set of initial states that give the valuation of the formula it computes. These sets, I, contain a (unique) state of the form (p, φ, c) for each control state p. In general, φ is the formula whose denotation is being computed, but, in the case of a fixed point, $\varphi = Z$ where Z is the variable bound by the fixed point. Hence, we introduce the notation

$$I(p) = (p, \varphi, c) \text{ where } (p, \varphi, c) \in I$$

to denote the valuation for a given control state p. For a control state p and characters a, b, for forward time modalities, let $Next(p, a) = \{\ (p', w)\ \mid p\,a \rightarrow p'\,w\ \}$, for backwards time, let $\overline{Pop}(p) = \{\ (p', a')\ \mid p'\,a' \rightarrow p\,\varepsilon\ \}$, and $\overline{Rew}(p, a) = \{\ (p', a')\ \mid p'\,a' \rightarrow p\,b\ \}$, $\overline{Push}(p, a, b) = \{\ (p', a')\ \mid p'\,a' \rightarrow p\,ab\ \}$, and together $Pre(p, a, b) = \overline{Pop}(p) \cup \overline{Rew}(p, a) \cup \overline{Push}(p, a, b)$. Most automaton states are of the form (p, φ, c) which represents a working value of the denotation of φ restricted to the control state p. The last element c is an integer that broadly corresponds to the fixed point depth of φ in χ. There are also states of the form (p, φ, c, a) which are used as intermediate states for the backwards time computations. For convenience we equate all $(p, \varphi, c, \bot)$ with q^{ε}_{f}.

We define the projection function

$$\pi_c(q) = \begin{cases} (p, \varphi, c+1) & \text{if } q = (p, \varphi, c) \\ (p, \varphi, c+1, a) & \text{if } q = (p, \varphi, c, a) \\ q & \text{otherwise} \end{cases}$$

which we lift to sets of states in the obvious way. This projection function can be compared with the projections discussed in Section 1.5. Here, the states $(p, \varphi, c+1)$ correspond to the new initial states, and (p, φ, c) to the old.

For an automaton A and variable Z, we say that the variable has the set of *binding states* (p, Z, c) for all control states p such that c is the largest value for which (p, Z, c) is in A. We say an automaton A gives a valuation of an environment if it contains an initial state $(p, \widehat{x}, *)$ for every atomic proposition and control state and a binding state for every free variable and control state, such that, for a given Z, all binding states have the same c. Let $\mathcal{Q}^{A}_{Z}$ be the set of binding states of Z in A and $\mathcal{Q}^{A}_{\widehat{x}}$ be the set of all $(p, \widehat{x}, *)$. In addition, let $level(p, \varphi, c) = c$ and $A[\varphi/I]$ be a renaming function on automata that renames states of the form $(p, \varphi', c) \in I$ to (p, φ, c). The sets I will be suitably defined to avoid name clashes.

We also assume that all automata have (share) the states q^* and q^{ε}_{f}, where q^{ε}_{f} is accepting and $q^* \xrightarrow{a} \{q^*\}$ for all $a \in \Sigma \setminus \{\bot\}$ and $q^* \xrightarrow{\bot} \left\{q^{\varepsilon}_{f}\right\}$. Furthermore, all transitions of the form $q \xrightarrow{\bot} Q$ have $Q = \left\{q^{\varepsilon}_{f}\right\}$. Finally, we introduce a comparison operator $A \preceq A'$, which can be intuitively read as $\mathcal{L}(A) \subseteq \mathcal{L}(A')$. The precise definition is deferred to Definition 3.2.

In Section 4 we give the pre- and post-conditions of each of the given procedures. Correctness is shown in Section 4.

Procedure 1 *Denotation*$(\chi, A_V, \mathbb{P})$

Require: A pushdown system $\mathbb{P} = (\mathcal{P}, \mathcal{D}, \Sigma)$, a modal μ-calculus formula χ and an automaton A_V giving valuations for all (unbound) literals.
Ensure: A pair (A, I) such that automaton A recognises $[\![\chi_1]\!]^{\mathbb{P}}_V$ from initial states I.

return $\mathit{Dispatch}(A_V, \chi, 1, \mathbb{P})$

Procedure 2 *Dispatch*$(A, \varphi, c, \mathbb{P})$

---

if $\varphi = \widehat{x}$ **then**
  return $(A, \mathcal{Q}^A_{\widehat{x}})$
else if $\varphi = Z$ **then**
  return $(A, \mathcal{Q}^A_Z)$
else if $\varphi = \varphi_1 \wedge \varphi_2$ **then**
  return $\mathit{And}(A, \varphi_1, \varphi_2, c, \mathbb{P})$
else if $\varphi = \varphi_1 \vee \varphi_2$ **then**
  return $\mathit{Or}(A, \varphi_1, \varphi_2, c, \mathbb{P})$
else if $\varphi = \Box\varphi_1$ **then**
  return $\mathit{Box}(A, \varphi_1, c, \mathbb{P})$
else if $\varphi = \Diamond\varphi_1$ **then**
  return $\mathit{Diamond}(A, \varphi_1, c, \mathbb{P})$
else if $\varphi = \overline{\Box}\varphi_1$ **then**
  return $\mathit{BackBox}(A, \varphi_1, c, \mathbb{P})$
else if $\varphi = \overline{\Diamond}\varphi_1$ **then**
  return $\mathit{BackDiamond}(A, \varphi_1, c, \mathbb{P})$
else if $\varphi = \mu Z.\varphi_1$ **then**
  return $\mathit{LFP}(A, Z, \varphi_1, c, \mathbb{P})$
else if $\varphi = \nu Z.\varphi_1$ **then**
  return $\mathit{GFP}(A, Z, \varphi_1, c, \mathbb{P})$
end if

---

3. Termination

3.1. Comparing Automata

We begin by defining the $\preceq$ operator described intuitively in Section 2. Observe that if we have $q \xrightarrow{a} Q$ and $q \xrightarrow{a} Q'$ with $Q \subseteq Q'$, then acceptance from Q' implies acceptance from Q. That is, the transition to Q' can, in some sense, be simulated by the transition to Q. Furthermore, acceptance from any q that is

Procedure 3 $And(A, \varphi_1, \varphi_2, c, \mathbb{P})$

$((\mathcal{Q}_1, \Sigma, \Delta_1, \_, \mathcal{F}_1), I_1) = Dispatch(A, \varphi_1, c, \mathbb{P})$
$((\mathcal{Q}_2, \Sigma, \Delta_2, \_, \mathcal{F}_2), I_2) = Dispatch(A, \varphi_2, c, \mathbb{P})$
$A' = (\mathcal{Q}_1 \cup \mathcal{Q}_2 \cup I, \Sigma, \Delta_1 \cup \Delta_2 \cup \Delta', \_, \mathcal{F}_1 \cup \mathcal{F}_2)$
where $I = \{\ (p, \varphi_1 \wedge \varphi_2, c)\ \mid p \in \mathcal{P}\ \}$
and $\Delta' = \left\{ ((p, \varphi_1 \wedge \varphi_2, c), a, Q_1 \cup Q_2) \;\middle|\; \begin{array}{c} (I_1(p), a, Q_1) \in \Delta_1 \wedge \\ (I_2(p), a, Q_2) \in \Delta_2 \end{array} \right\}$
return (A', I)

Procedure 4 $Or(A, \varphi_1, \varphi_2, c, \mathbb{P})$

$((\mathcal{Q}_1, \Sigma, \Delta_1, \_, \mathcal{F}_1), I_1) = Dispatch(A, \varphi_1, c, \mathbb{P})$
$((\mathcal{Q}_2, \Sigma, \Delta_2, \_, \mathcal{F}_2), I_2) = Dispatch(A, \varphi_2, c, \mathbb{P})$
$A' = (\mathcal{Q}_1 \cup \mathcal{Q}_2 \cup I, \Sigma, \Delta_1 \cup \Delta_2 \cup \Delta', \_, \mathcal{F}_1 \cup \mathcal{F}_2)$
where $I = \{\ (p, \varphi_1 \vee \varphi_2, c)\ \mid p \in \mathcal{P}\ \}$
and $\Delta' = \left\{ ((p, \varphi_1 \vee \varphi_2, c), a, Q) \;\middle|\; \begin{array}{c} (I_1(p), a, Q) \in \Delta_1 \vee \\ (I_2(p), a, Q) \in \Delta_2 \end{array} \right\}$
return (A', I)

Procedure 5 $Box(A, \varphi_1, c, \mathbb{P})$

$((\mathcal{Q}_1, \Sigma, \Delta_1, \_, \mathcal{F}_1), I_1) = Dispatch(A, \varphi_1, c, \mathbb{P})$
$A' = (\mathcal{Q}_1 \cup I, \Sigma, \Delta_1 \cup \Delta', \_, \mathcal{F}_1)$
where $I = \{\ (p, \Box\varphi_1, c)\ \mid p \in \mathcal{P}\ \}$
and $\Delta' = \left\{ ((p, \Box\varphi_1, c), a, Q) \;\middle|\; \begin{array}{c} Next(p, a) = \{(p_1, w_1), \ldots, (p_n, w_n)\} \wedge \\ \bigwedge_{1 \le j \le n} \left(I_1(p_j) \xrightarrow[\Delta_1]{w_j} Q_j \right) \wedge \\ Q = Q_1 \cup \cdots \cup Q_n \end{array} \right\} \cup$
$\{\ ((p, \Box\varphi_1, c), a, \{q^*\})\ \mid Next(p, a) = \emptyset \wedge a \neq \perp\ \} \cup$
$\left\{\ \left((p, \Box\varphi_1, c), \perp, \left\{q_f^\varepsilon\right\}\right)\ \mid Next(p, \perp) = \emptyset\ \right\}$
return (A', I)

Procedure 6 $Diamond(A, \varphi_1, c, \mathbb{P})$

$((\mathcal{Q}_1, \Sigma, \Delta_1, \_, \mathcal{F}_1), I_1) = Dispatch(A, \varphi_1, c, \mathbb{P})$
$A' = (\mathcal{Q}_1 \cup I, \Sigma, \Delta_1 \cup \Delta', \_, \mathcal{F}_1)$
where $I = \{\ (p, \Diamond\varphi_1, c)\ \mid p \in \mathcal{P}\ \}$
and $\Delta' = \left\{ ((p, \Diamond\varphi_1, c), a, Q) \;\middle|\; \begin{array}{c} (p', w) \in Next(p, a) \wedge \\ I_1(p') \xrightarrow[\Delta_1]{w} Q \end{array} \right\}$
return (A', I)

Procedure 7 $BackBox(A, \varphi_1, c, \mathbb{P})$

$((\mathcal{Q}_1, \Sigma, \Delta_1, \_, \mathcal{F}_1), I_1) = Dispatch(A, \varphi_1, c, \mathbb{P})$
$A' = (\mathcal{Q}_1 \cup I \cup \mathcal{Q}_{int}, \Sigma, \Delta_1 \cup \Delta', \_, \mathcal{F}_1)$
where $I = \left\{ \ (p, \overline{\Box}\varphi_1, c) \ \mid p \in \mathcal{P} \ \right\}$
and $\mathcal{Q}_{int} = \left\{ \ (p, \overline{\Box}\varphi_1, c, a) \ \mid p \in \mathcal{P} \wedge a \in \Sigma \ \right\}$
and $\Delta' =$

$$\left\{ ((p, \overline{\Box}\varphi_1, c), a, Q) \ \middle| \ \begin{array}{c} Q = \left\{(p, \overline{\Box}\varphi_1, c, a)\right\} \cup Q_{pop} \cup Q_{rew} \ \wedge \\ \overline{Pop}(p) = \{(p_1, a_1), \ldots, (p_n, a_n)\} \ \wedge \\ \bigwedge_{1 \leq j \leq n} \left(I_1(p_j) \xrightarrow[\Delta_1]{a_j} Q'_j \xrightarrow[\Delta_1]{a} Q_j^{pop} \right) \ \wedge \\ Q_{pop} = Q_1^{pop} \cup \cdots \cup Q_n^{pop} \ \wedge \\ \overline{Rew}(p, a) = \{(p'_1, a'_1), \ldots, (p'_{n'}, a'_{n'})\} \ \wedge \\ \bigwedge_{1 \leq j \leq n'} \left(I_1(p'_j) \xrightarrow[\Delta_1]{a'_j} Q_j^{rew} \right) \ \wedge \\ Q_{rew} = Q_1^{rew} \cup \cdots \cup Q_n^{rew} \end{array} \right\} \cup$$

$$\left\{ ((p, \overline{\Box}\varphi_1, c, a), b, Q) \ \middle| \ \begin{array}{c} Pre(p, a, b) = \{(p_1, a_1), \ldots, (p_n, a_n)\} \wedge \\ \bigwedge_{1 \leq j \leq n} \left(I_1(p_j) \xrightarrow[\Delta_1]{a_j} Q_j^{push} \right) \wedge \\ Q = Q_1^{push} \cup \cdots \cup Q_n^{push} \end{array} \right\} \cup$$

$$\left\{ \ ((p, \overline{\Box}\varphi_1, c), a, \{q^*\}) \ \mid \forall b. Pre(p, a, b) = \emptyset \ \right\} \cup$$
$$\left\{ \ \left((p, \overline{\Box}\varphi_1, c), \bot, \left\{ q_f^{\varepsilon} \right\} \right) \ \mid \forall a. Pre(p, \bot, a) = \emptyset \ \right\} \cup$$
$$\left\{ \ ((p, \overline{\Box}\varphi_1, c, a), b, \{q^*\}) \ \mid \overline{Push}(p, a, b) = \emptyset \ \right\} \cup$$
$$\left\{ \ \left((p, \overline{\Box}\varphi_1, c, a), \bot, \left\{ q_f^{\varepsilon} \right\} \right) \ \mid \overline{Push}(p, a, \bot) = \emptyset \ \right\}$$

return (A', I)

Procedure 8 $BackDiamond(A, \varphi_1, c, \mathbb{P})$

$((\mathcal{Q}_1, \Sigma, \Delta_1, \_, \mathcal{F}_1), I_1) = Dispatch(A, \varphi_1, c, \mathbb{P})$
$A' = (\mathcal{Q}_1 \cup I \cup \mathcal{Q}_{int}, \Sigma, \Delta_1 \cup \Delta', \_, \mathcal{F}_1)$
where $I = \left\{ \ (p, \overline{\Diamond}\varphi_1, c) \ \mid p \in \mathcal{P} \ \right\}$
and $\mathcal{Q}_{int} = \left\{ \ (p, \overline{\Box}\varphi_1, c, a) \ \mid p \in \mathcal{P} \wedge a \in \Sigma \ \right\}$

and $\Delta' =$
$$\begin{array}{c} \left\{ ((p, \overline{\Diamond}\varphi_1, c), a, Q) \ \middle| \ \begin{array}{c} (p', a') \in \overline{Pop}(p) \wedge \\ I_1(p') \xrightarrow[\Delta_1]{a'} Q' \xrightarrow[\Delta_1]{a} Q \end{array} \right\} \cup \\ \left\{ ((p, \overline{\Diamond}\varphi_1, c), a, Q) \ \middle| \ \begin{array}{c} (p', a') \in \overline{Rew}(p, a) \wedge \\ I_1(p') \xrightarrow[\Delta_1]{a'} Q \end{array} \right\} \cup \\ \left\{ ((p, \overline{\Diamond}\varphi_1, c), a, \left\{(p, \overline{\Diamond}\varphi_1, c, a)\right\}) \right\} \cup \\ \left\{ ((p, \overline{\Diamond}\varphi_1, c, a), b, Q) \ \middle| \ \begin{array}{c} (p', a') \in \overline{Push}(p, a, b) \wedge \\ I_1(p') \xrightarrow[\Delta_1]{a'} Q \end{array} \right\} \end{array} .$$

return (A', I)

Procedure 9 $LFP(A, Z, \varphi_1, c, \mathbb{P})$

$A_0 = (\mathcal{Q} \cup I_c, \Sigma, \Delta, \_, \mathcal{F})$
where $I_c = \{\ (p, Z, c)\ \mid p \in \mathcal{P}\ \}$
for $i = 0$ to ω **do**
  $(B_i, I_i) = Dispatch(A_i, \varphi_1, c+1, \mathbb{P})$
  $A_{i+1} = Proj(B_i[Z/I_i], c)$
  if $A_{i+1} \preceq A_i$ **then**
    return (A_i, I_c)
  end if
end for

Procedure 10 $GFP(A, Z, \varphi_1, c, \mathbb{P})$

$A_0 = (\mathcal{Q} \cup I_c, \Sigma, \Delta \cup \Delta', \_, \mathcal{F})$
where $I_c = \{\ (p, Z, c)\ \mid p \in \mathcal{P}\ \}$
and Δ' contains $q \xrightarrow{a} \{q^*\}$ for all $a \neq \perp$ and $q \xrightarrow{\perp} \left\{q_f^\varepsilon\right\}$ for all $q \in I_c$.
for $i = 0$ to ω **do**
  $(B_i, I_i) = Dispatch(A_i, \varphi_1, c+1, \mathbb{P})$
  $A_{i+1} = Proj(B_i[Z/I_i], c)$
  if $A_i \preceq A_{i+1}$ **then**
    return (A_i, I_c)
  end if
end for

Procedure 11 $Proj(A, c)$

$A' = A$
for all q with $level(q) = c+1$ **do**
  Replace each transition $q \xrightarrow{a} Q$ in A' with $q \xrightarrow{a} \pi_c(Q)$.
end for
for all q with $level(q) = c$ **do**
  Remove q from A'.
end for
for all $q = (p, \varphi', c+1)$ in A' for some p and φ' **do**
  Rename q to (p, φ', c).
end for
return A'

not q^ε_f implies acceptance from q^* (trivially). Using these observations, we can provide a simple test implying that $\mathcal{L}(A) \subseteq \mathcal{L}(A')$. In the following definition, $Q \ll Q'$ can be taken to mean an accepting run from Q' implies an accepting run from Q.

Definition 3.1. *For all non-empty sets of states Q and Q', we define*

$$Q \ll Q' \;:=\; \big((q^* \in Q \Rightarrow \exists q. q \neq q^\varepsilon_f \wedge q \in Q') \;\wedge\; (\forall q \neq q^*. q \in Q \Rightarrow q \in Q')\big)$$

We define $\preceq$ by extending this definition to automata as follows.

Definition 3.2. *For automata A and A' with state-sets $\mathcal{Q}$ and $\mathcal{Q}'$ respectively, we define $A \preceq A'$ just if for all $q \in \mathcal{Q} \cap \mathcal{Q}'$, a and Q, if $q \xrightarrow[A]{a} Q$ then for some Q', $q \xrightarrow[A']{a} Q'$ and $Q' \ll Q$.*

By induction, $\preceq$ can be applied to full runs. Observe that this implies, for each shared state q, $\mathcal{L}_q(A) \subseteq \mathcal{L}_q(A')$. Since A and A' need not share the same state set, one of the consequences of using $\ll$ is that q^* can take the place of a state that is not shared between the automata. This is important after the first iteration of the greatest fixed point computations, since the recursive call may add states that were not in the initial automaton A_0.

Lemma 3.1. *For automata A and A' with state-sets $\mathcal{Q}$ and $\mathcal{Q}'$ respectively, if $A \preceq A'$ then for all $q \in \mathcal{Q} \cap \mathcal{Q}'$, w and Q, if $q \xrightarrow[A]{w} Q$ then for some Q', $q \xrightarrow[A']{w} Q'$ and $Q' \ll Q$.*

Proof. We prove for all $Q_1 \subseteq \mathcal{Q}$, $Q_2 \subseteq \mathcal{Q}'$ and w that, if $Q_2 \ll Q_1$ and $Q_1 \xrightarrow[A]{w} Q'_1$ for some Q'_1, then there exists Q'_2 such that $Q_2 \xrightarrow[A']{w} Q'_2$ and $Q'_2 \ll Q'_1$. We proceed by induction over the length of w.

Let $Q_1 = \{q^1_1, \ldots, q^1_n\}$. We have that $q^1_i \xrightarrow[A]{a} Q^i_1$ for all $1 \leq i \leq n$ and $Q'_1 = Q^1_1 \cup \cdots \cup Q^n_1$. When $a = \perp$, the property is immediate from $A \preceq A'$ and the assumed format of $\perp$-transitions. Otherwise $a \neq \perp$ and for each q^1_i there are two cases. Either $q^* \in Q_2$ or $q^1_i \in Q_2$. In the first case, we have $q^* \xrightarrow[A']{a} Q^i_2$ where $Q^i_2 = \{q^*\}$, and hence $Q^i_2 \ll Q^i_1$. In the second case, we have, from $A \preceq A'$ some transition $q^1_i \xrightarrow[A']{a} Q^i_2$ with $Q^i_2 \ll Q^i_1$. Thus, we have $Q'_2 = Q^1_2 \cup \cdots \cup Q^n_2 \ll Q^1_1 \cup \cdots \cup Q^n_1 = Q'_1$ as required. This concludes the base case.

Inductively, assume $w = aw'$ and a run $Q_1 \xrightarrow[A]{a} Q''_1 \xrightarrow[A]{w'} Q'_1$. By repeating the above argument we have $Q_2 \xrightarrow[A']{a} Q''_2$ with $Q''_2 \ll Q''_1$. Then, by induction over the length of the run we have $Q''_2 \xrightarrow[A']{w'} Q'_2$ with $Q'_2 \ll Q'_1$. This gives us the required run over aw. □

To prove termination, we will require the notion of an expansion.

Definition 3.3. *Given an automaton A with state-set $\mathcal{Q}$, we define*

$$\textsc{Expand}(A) := \left\{ \ q \xrightarrow{a} Q' \ \middle| \ q \xrightarrow{a} Q \ \textit{in } A \textit{ and } Q \ll Q' \subseteq \mathcal{Q} \ \right\} \ .$$

To test termination of the fixed point computations, we compare $\textsc{Expand}(A_{i+1})$ and $\textsc{Expand}(A_i)$. In the following proofs we assume both automata share the same state-set.

Lemma 3.2. $\textsc{Expand}(A) \subseteq \textsc{Expand}(A')$ *if and only if* $A \preceq A'$.

Proof. First we assume $\textsc{Expand}(A) \subseteq \textsc{Expand}(A')$. Take $q \xrightarrow{a} Q$ in A. Then $q \xrightarrow{a} Q \in \textsc{Expand}(A)$. We have $q \xrightarrow{a} Q \in \textsc{Expand}(A')$, and therefore $q \xrightarrow{a} Q'$ is a transition of A' with $Q' \ll Q$.

In the other direction, we assume $q \xrightarrow{a} Q$ in A implies $q \xrightarrow{a} Q'$ in A'. Take $q \xrightarrow{a} Q \in \textsc{Expand}(A)$. We need $q \xrightarrow{a} Q \in \textsc{Expand}(A')$. We have some $q \xrightarrow{a} Q'$ in A with $Q' \ll Q$. Hence, we have $q \xrightarrow{a} Q''$ in A' with $Q'' \ll Q$. Hence, $q \xrightarrow{a} Q \in \textsc{Expand}(A')$ as required. □

We extend the property to runs. Hence $\textsc{Expand}(A) \subseteq \textsc{Expand}(A')$ implies $\mathcal{L}(A) \subseteq \mathcal{L}(A')$.

Lemma 3.3. *If* $\textsc{Expand}(A) \subseteq \textsc{Expand}(A')$ *then whenever* $q \xrightarrow{w} Q$ *in* A *then there is some* $Q' \ll Q$ *with* $q \xrightarrow{w} Q'$ *in* A'.

Proof. This follows directly from Lemma 3.2 and Lemma 3.1. □

3.2. Algorithm Termination

We prove the following to show termination.

Lemma 3.4 (Termination). *The algorithm satisfies the following properties.*

1. *Each subroutine introduces a fixed set of new states, independent of the automaton A given as input (but may depend on the other parameters). Transitions are only added to these new states.*
2. *For two input automata A_1 and A_2 (giving valuations of the same environments) such that $A_1 \preceq A_2$, then the returned automata A'_1 and A'_2, respectively, satisfy $A'_1 \preceq A'_2$.*
3. *The algorithm terminates.*

Proof. The first of these conditions is trivially satisfied by all constructions, hence we omit the proofs. Similarly, termination is trivial for all procedures except the fixed point constructions. We will say a procedure is *monotonic* if is satisfies the second condition. The second and third conditions will be show by mutual induction over the recursion (structure of the formula). The cases $\widehat{x}$ and Z are immediate.

Case $\mathit{And}(A, \varphi_1, \varphi_2, c, \mathbb{P})$:

Take $A \preceq A'$ both giving valuations for V. After the recursive calls we have $A_1 \preceq A'_1$ and $A_2 \preceq A'_2$. New transitions are only added to new states, which are the same in A_1 and A'_1 (as part of the termination conditions), and similarly for A_2 and A'_2. Let the results for the intersection be $A_\wedge$ and $A'_\wedge$ respectively. For all p we have $(p, \varphi_1 \wedge \varphi_2, c) \xrightarrow[A_\wedge]{a} Q$ derived from $I_1(p) \xrightarrow[A_1]{a} Q_1$ and $I_2(p) \xrightarrow[A_2]{a} Q_2$. Hence we have $I_1(p) \xrightarrow[A'_1]{a} Q'_1$ and $I_2(p) \xrightarrow[A'_2]{a} Q'_2$ and thus $(p, \varphi_1 \wedge \varphi_2, c) \xrightarrow[A'_\wedge]{a} Q'$ such that $Q' = Q'_1 \cup Q'_2 \ll Q_1 \cup Q_2 = Q$ as required.

Case $\mathit{Or}(A, \varphi_1, \varphi_2, c, \mathbb{P})$:

Take $A \preceq A'$ both giving valuations for V. After the recursive calls we have $A_1 \preceq A'_1$ and $A_2 \preceq A'_2$. New transitions are only added to new states, which are the same in A_1 and A'_1 (as part of the termination conditions), and similarly for A_2 and A'_2. Let the results for the disjunction be $A_\vee$ and $A'_\vee$ respectively. For all p we have $(p, \varphi_1 \vee \varphi_2, c) \xrightarrow[A_\vee]{a} Q$ derived from $I_1(p) \xrightarrow[A_1]{a} Q$ or $I_2(p) \xrightarrow[A_2]{a} Q$. Hence we have $I_1(p) \xrightarrow[A'_1]{a} Q'$ or $I_2(p) \xrightarrow[A'_2]{a} Q'$ and thus $(p, \varphi_1 \vee \varphi_2, c) \xrightarrow[A'_\vee]{a} Q'$ such that $Q' \ll Q$ as required.

Case $\mathit{Box}(A, \varphi_1, c, \mathbb{P})$:

Take $A \preceq A'$ both giving valuations for V. After the recursive calls we have $A_1 \preceq A'_1$. New transitions are only added to new states, which are the same in A_1 and A'_1 (as part of the termination conditions). Let the results for the box be $A_\Box$ and $A'_\Box$ respectively. Take a new transition $(p, \Box\varphi_1, c) \xrightarrow[A_\Box]{a} Q$. Since the case when $\mathit{Next}(p, a) = \emptyset$ is immediate, let $\mathit{Next}(p, a) = \{(p_1, w_1), \ldots, (p_n, w_n)\}$. We have $Q = Q_1 \cup \cdots \cup Q_n$ where for each $1 \leq i \leq n$ we have $I_1(p_i) \xrightarrow[A_1]{w_i} Q_i$. By $A_1 \preceq A'_1$ we have $I_1(p_i) \xrightarrow[A'_1]{w_i} Q'_i$ with $Q_i \ll Q'_i$. Hence, we have $(p, \Box\varphi_1, c) \xrightarrow[A'_\Box]{a} Q'$ with $Q' = Q'_1 \cup \cdots \cup Q'_n \ll Q_1 \cup \cdots Q_n = Q$ as required.

Case $\mathit{Diamond}(A, \varphi_1, c, \mathbb{P})$:

Take $A \preceq A'$ both giving valuations for V. After the recursive calls we have $A_1 \preceq A'_1$. New transitions are only added to new states, which are the same in A_1 and A'_1 (as part of the termination conditions). Let the results for the box be $A_\Diamond$ and $A'_\Diamond$ respectively. Take a new transition $(p, \Diamond\varphi_1, c) \xrightarrow[A_\Diamond]{a} Q$. Take some $(p', w') \in \mathit{Next}(p, a)$. We have $I_1(p') \xrightarrow[A_1]{w'} Q$. By $A_1 \preceq A'_1$ we have $I(p') \xrightarrow[A'_1]{w'} Q'$ with $Q' \ll Q$. Hence, we have $(p, \Diamond\varphi_1, c) \xrightarrow[A'_\Diamond]{a} Q'$ with $Q' \ll Q$ as required.

Case $\mathit{BackBox}(A, \varphi_1, c, \mathbb{P})$ **and** $\mathit{BackDiamond}(A, \varphi_1, c, \mathbb{P})$:

It can be observed that all new transitions in A are derived from transitions $I(p') \xrightarrow[A]{a} Q$ (or are independent of A and A'). Since $A \preceq A'$ it follows that all transitions have a counterpart $I(p') \xrightarrow[A']{a} Q'$ with $Q' \ll Q$. Hence the property follows in a similar manner to the previous cases.

Case $LFP(A, Z, \varphi_1, c, \mathbb{P})$:

Note that the state-set of A_0 is a subset of the states of A_1 (since it does not contain the states introduced by the recursive call). However, for all $i \geq 1$, all A_i have the same states. Initially we have $A_0 \preceq A_1$ since the shared states of A_0 and A_1 are either given by A (and hence have the same transitions), or have no transitions in A_0. Since the recursive call is monotonic, and the projections do not affect monotonicity, we have by induction that $A_i \preceq A_{i+1}$ for all i. For all $i \geq 1$, we have by Lemma 3.2 that $\textsc{Expand}(A_i) \subseteq \textsc{Expand}(A_{i+1})$. Since the set of states is fixed, we must eventually have $\textsc{Expand}(A_i) = \textsc{Expand}(A_{i+1})$ and hence $A_{i+1} \preceq A_i$, resulting in termination.

Monotonicity follows directly from the monotonicity of the recursive call, and that the projections do not affect the monotonicity property.

Case $GFP(A, Z, \varphi_1, c, \mathbb{P})$:

Note that the state-set of A_0 is a subset of the states of A_1 (since it does not contain the states introduced by the recursive call). However, for all $i \geq 1$, all A_i have the same states. Initially we have $A_1 \preceq A_0$ since the shared states of A_0 and A_1 are either given by A (and hence have the same transitions), or have transitions to q^* or q_f^ε that always imply $\ll$ as required. Since the recursive call is monotonic, and the projections do not affect monotonicity, we have by induction that $A_{i+1} \preceq A_i$ for all i. For all $i \geq 1$, we have by Lemma 3.2 that $\textsc{Expand}(A_{i+1}) \subseteq \textsc{Expand}(A_i)$. Since the set of states is fixed, we must eventually have $\textsc{Expand}(A_i) = \textsc{Expand}(A_{i+1})$ and hence $A_i \preceq A_{i+1}$, resulting in termination.

Monotonicity follows directly from the monotonicity of the recursive call, and that the projections do not affect the monotonicity property. $\square$

3.3. Complexity

The algorithm runs in EXPTIME. Let m be the nesting depth of the fixed points of the formula and n be the number of states in A_V. We introduce at most $k = \mathcal{O}(|\mathcal{P}| \cdot |\chi| \cdot m \cdot |\Sigma|)$ states to the automaton. Hence, there are at most $\mathcal{O}(n+k)$ states in the automaton during any stage of the algorithm. The fixed point computations iterate up to an $\mathcal{O}\left(2^{\mathcal{O}(n+k)}\right)$ number of times. Each iteration has a recursive call, which takes up to $\mathcal{O}\left(2^{\mathcal{O}(n+k)}\right)$ time. Hence the algorithm is $\mathcal{O}\left(2^{\mathcal{O}(n+k)}\right)$ overall.

4. Correctness

4.1. Valuation Soundness and Completeness

To prove correctness, we will introduce the notion of a *valuation profile*, which is a mapping $V : \mathcal{Q} \to \Sigma^*\bot$. Intuitively, a valuation profile maps each state of a automaton to a set of words that should be accepted from that state. For example, $V(q^*) = \Sigma^*\bot$ since all valid stacks are accepted from q^*. Similarly, $V(q^{\varepsilon}_f) = \{\varepsilon\}$. Note that we overload V to represent valuation profiles and modal μ-calculus valuations. It will be clear from the context which usage is intended.

Given a valuation profile V and some c, we can extract a modal μ-calculus valuation V_c as follows. Let $V_c(Z) = \{ \langle p, w\rangle \mid w \in V(p, Z, c') \}$ where c' is the largest $c' \leq c$ such that $V(p, Z, c')$ is defined.

We introduce *valuation soundness* and *valuation completeness* based on a profile V. We prove that all subroutines of the algorithm have this property. First, it is worth taking some time to understand the benefits of valuation soundness and completeness in proving the correctness of the algorithm.

The main challenge in proving correctness is to show that the projections do not cause any violations to correctness: the rest of the algorithm can be seen, rather straightforwardly, to be correct. Given a transition from some state $(p, \varphi, c + 1)$ to a set of states Q, the effect of the projections is to replace every occurrence of (p, φ, c) in Q with $(p, \varphi, c + 1)$. Valuation soundness and completeness formalises the intuition that these two states represent two working values of the same denotation. Hence, replacing one with the other will maintain correctness.

More precisely *valuation soundness* captures the observation that the existence of an a-transition in an automaton means that the a character can be prepended to any word accepted by the destination of the transition. For an automaton to be valuation sound with respect to some V, then all of its transitions must be in accordance with V.

Definition 4.1. *Given a valuation V, an automaton A is V-sound just if, for all q, a and w, if A has a transition $q \xrightarrow{a} Q$ such that $w \in V(q')$ for all $q' \in Q$, then $a\,w \in V(q)$.*

By induction on the length of the word, valuation soundness extends to runs of an automaton. We then obtain that all accepting runs are sound.

Lemma 4.1. *Let A be a V-sound automaton.*

1. *For all q, w and w', if A has a run $q \xrightarrow{w} Q$ such that $w' \in V(q')$ for all $q' \in Q$, then $w\,w' \in V(q)$.*
2. *For all $q \in \mathcal{Q}_A$, $\mathcal{L}_q(A) \subseteq V(q)$.*

Proof. (i) We prove by induction on the length of the word w. When $w = a$, the property is just V-soundness. Take $w = au$ and some run $q \xrightarrow{a} Q \xrightarrow{u} Q'$ such that for all $q' \in Q'$, we have $w \in V(q')$. By the induction hypothesis, we have the property for the run $Q \xrightarrow{u} Q'$. Hence, we have for all $q' \in Q$ that, $uw' \in V(q')$. Thus, from V-soundness, we have $auw' \in V(q)$.

(ii) Take an accepting run $q \xrightarrow{w} Q_f$ of A. We have for all $q' \in Q_f = \left\{q^{\varepsilon}_f\right\}$, $\varepsilon \in V(q')$. Thanks to (i), we have $w \in V(q)$. □

Valuation completeness is the dual notion to valuation soundness. It says that if a character can begin a word that should be accepted from a given state, then there should be a transition that witnesses this. Furthermore, the transition should be in accordance with the given valuation V.

Definition 4.2. *Given a valuation V, an automaton A is V-complete just if, for all q, a and w, if $a\,w \in V(q)$ then A has a transition $q \xrightarrow{a} Q$ such that $w \in V(q')$ for all $q' \in Q$.*

By induction on the length of the word, valuation completeness extends to runs. Furthermore, an accepting run always exists when required.

Lemma 4.2. *Let A be a V-complete automaton.*

1. *For all q, w and w', if $w\,w' \in V(q)$ then A has a run $q \xrightarrow{w} Q$ such that $w' \in V(q')$ for all $q' \in Q$.*
2. *For all $q \in \mathcal{Q}_A$, $V(q) \subseteq \mathcal{L}_q(A)$.*

Proof. (i) The proof is by induction on the length of the word w. When $w = a$, the property is simply V-completeness. Take $w = au$ and some q with $auw' \in V(q)$. From V-completeness, we have a transition $q \xrightarrow{a} Q$ such that for all $q' \in Q$, we have $uw \in V(q')$. By induction on the length of the word, we have a run $Q \xrightarrow{u} Q'$ satisfying the property. Hence, we have $q \xrightarrow{a} Q \xrightarrow{u} Q'$ as required.

(ii) Take $w \in V(q)$. Instantiating (i) with $w' = \varepsilon$, we know A has a run $q \xrightarrow{w} Q$. Every state in Q must be accepting because ε is only accepted from accepting states and there can be no $\langle p^j, \varepsilon\rangle$ satisfying any denotation because ε is not a valid stack. $\square$

4.2. Algorithm Correctness

To define the correctness conditions we need to define the extension of a valuation profile by a formula φ. For a variable Z bound in φ, we denote by φ_Z the sub-formula of φ that binds Z.

Definition 4.3. *Given a valuation profile V, we define V^c_φ for a given c and φ such that for sub-formulas φ' of φ*

$$V^c_\varphi(p, \varphi', c') = \begin{cases} V(p, \varphi', c') & \text{if } c' < c \\ \left\{ w \;\middle|\; \langle p, w\rangle \in [\![\varphi_Z]\!]^{\mathbb{P}}_{(V^c_\varphi)_c} \right\} & \text{if } \varphi' = Z \text{ and } c' = c \\ \left\{ w \;\middle|\; \langle p, w\rangle \in [\![\varphi']\!]^{\mathbb{P}}_{(V^c_\varphi)_c} \right\} & \text{otherwise} \end{cases}$$

and

$$V^c_\varphi(p, \overline{\Box}\varphi', c', a) = \begin{cases} V(p, \overline{\Box}\varphi', c', a) & \text{if } c' < c \\ \left\{ bw \;\middle|\; \begin{array}{r} \forall \langle p', a'w\rangle \hookrightarrow \langle p, abw\rangle . \\ \langle p', a'w\rangle \in [\![\varphi']\!]^{\mathbb{P}}_{(V^c_\varphi)_c} \end{array} \right\} & \text{otherwise} \end{cases}$$

and

$$V^c_\varphi(p, \overline{\Diamond}\varphi', c', a) = \left\{ \begin{array}{ll} V(p, \overline{\Diamond}\varphi', c', a) & \textit{if } c' < c \\ \left\{ bw \;\middle|\; \begin{array}{r} \exists\langle p', a'w\rangle \hookrightarrow \langle p, abw\rangle. \\ \langle p', a'w\rangle \in [\![\varphi']\!]^{\mathbb{P}}_{(V^c_\varphi)_c} \end{array} \right\} & \textit{otherwise} \end{array} \right. .$$

(Note that this definition is circular. The definition can be made recursive by valuing the variables in order of alternation depth.)

We are now ready to state the correctness conditions.

Definition 4.4 (Correctness Conditions). *The correctness conditions are as follows. Let A be the input automaton, φ be the input formula*[1]*, c be the input level and A' be the result.*

1. *We only introduce level c states.*
2. *If A is V-sound, A' is V^c_φ-sound.*
3. *If A is V-complete, A' is V^c_φ-complete.*

We say that a procedure is V-sound/complete if the second/third condition is satisfied. That each procedure only introduces level c states is straightforward, hence we only show V-soundness and -completeness.

Lemma 4.3 (Valuation Soundness). *The algorithm is V-sound.*

Proof. The proof is by induction over the recursion. The base cases $\widehat{x}$ and Z are immediate.

Case $And(A, \varphi_1, \varphi_2, c, \mathbb{P})$:

By assumption, A is valuation sound with respect to some V. Furthermore, by induction, A_1 and A_2 are valuation sound with respect to $V^c_{\varphi_1}$ and $V^c_{\varphi_2}$ respectively.

We claim A' is sound with respect to $V^c_{\varphi_1 \wedge \varphi_2}$. This only has to be shown for the new transitions $((p, \varphi_1 \wedge \varphi_2, c), a, Q_1 \cup Q_2)$ derived from $(I_1(p), a, Q_1)$ and $(I_2(p), a, Q_2)$. Suppose some w such that for all $q \in Q_1 \cup Q_2$, $w \in V^c_{\varphi_1 \wedge \varphi_2}(q)$. Then, we have $w \in V^c_{\varphi_1}(q)$ and $w \in V^c_{\varphi_2}(q)$. Since A_1 and A_2 are sound, this implies $aw \in V^c_{\varphi_1}(I_1(p))$ and $aw \in V^c_{\varphi_2}(I_2(p))$ and hence $aw \in V^c_{\varphi_1 \wedge \varphi_2}(p, \varphi_1 \wedge \varphi_2, c)$ as required.

Case $Or(A, \varphi_1, \varphi_2, c, \mathbb{P})$:

By assumption, A is valuation sound with respect to some V. Furthermore, by induction, A_1 and A_2 are valuation sound with respect to $V^c_{\varphi_1}$ and $V^c_{\varphi_2}$ respectively.

We claim A' is sound with respect to $V^c_{\varphi_1 \vee \varphi_2}(Z, c')$. This only has to be shown for the new transitions $((p, \varphi_1 \vee \varphi_2, c), a, Q)$ derived from $(I_1(p), a, Q)$

[1]For cases such as $And(A, \varphi_1, \varphi_2, c, \mathbb{P})$ we take, as appropriate $\varphi = \varphi_1 \wedge \varphi_2$.

or $(I_2(p), a, Q)$. By symmetry, we only handle the first case. Suppose some w such that for all $q \in Q$, $w \in V^c_{\varphi_1 \vee \varphi_2}(q)$. Then, we have $w \in V^c_{\varphi_1}(q)$. Since A_1 is sound, this implies $aw \in V^c_{\varphi_1}(I_1(p))$ and hence $aw \in V^c_{\varphi_1 \vee \varphi_2}(p, \varphi_1 \vee \varphi_2, c)$ as required.

Case $Box(A, \varphi_1, c, \mathbb{P})$:

We assume that A is valuation sound with respect to some valuation V. By induction A_1 is valuation sound with respect to $V^c_{\varphi_1}$. We show that A' is valuation sound with respect to $V^c_{\Box\varphi_1}$.

We first deal with the case when $Next(p, a) = \emptyset$. In this case, the valuation of $\Box\varphi_1$ contains all words of the form aw for some w. Hence, all added transitions are trivially sound.

Otherwise, take a new transition $((p, \Box\varphi_1, c), a, Q)$ derived from the value of $Next(p, a) = \{(p_1, w_1), \ldots, (p_n, w_n)\}$ and for all $1 \leq j \leq n$, the runs $I_1(p_j) \xrightarrow[A_1]{w_j} Q_j$, with $Q = Q_1 \cup Q_n$. Suppose for some w, $w \in V^c_{\Box\varphi_1}(q)$ for all $q \in Q$. By valuation soundness of A_1 we know $w_j w \in V^c_{\Box\varphi_1}(I_1(p_j))$ and hence, since all transitions from $\langle p, aw \rangle$ lead to configurations satisfying φ_1, $aw \in V^c_{\Box\varphi_1}(p, \Box\varphi_1, c)$ as required.

Case $Diamond(A, \varphi_1, c, \mathbb{P})$:

We assume that A is valuation sound with respect to some valuation V. By induction A_1 is valuation sound with respect to $V^c_{\varphi_1}$. We show that A' is valuation sound with respect to $V^c_{\Diamond\varphi_1}$.

Take a new transition $((p, \Diamond\varphi_1, c), a, Q)$ derived from some $(p', w') \in Next(p, a)$ and the run $I_1(p') \xrightarrow[A_1]{w'} Q$. Suppose for some w, $w \in V^c_{\Diamond\varphi_1}(q)$ for all $q \in Q$. By valuation soundness of A_1 we know $w'w \in V^c_{\Diamond\varphi_1}(I_1(p'))$ and hence, since there is a transition from $\langle p, aw \rangle$ to a configuration satisfying φ_1, $aw \in V^c_{\Diamond\varphi_1}(p, \Diamond\varphi_1, c)$ as required.

Case $BackBox(A, \varphi_1, c, \mathbb{P})$:

We assume that A is valuation sound with respect to some valuation V. By induction A_1 is valuation sound with respect to $V^c_{\varphi_1}$. We show that A' is valuation sound with respect to $V^c_{\overline{\Box}\varphi_1}$.

We observe that no $(p', \overline{\Box}\varphi_1, c)$ are reachable from a state $(p, \overline{\Box}\varphi, c, a)$, hence we show soundness for the latter states first.

The first case is for some b with $\overline{Push}(p, a, b) = \emptyset$. In this case, the valuation of $(p, \overline{\Box}\varphi, c, a)$ contains all words of the form bw. Hence soundness is immediately satisfied.

Otherwise, $\overline{Push}(p, a, b) = \{(p_1, a_1), \ldots, (p_n, a_n)\}$ such that for all $1 \leq j \leq n$, $\langle p_j, a_i w \rangle \hookrightarrow \langle p, abw \rangle$. Take a new transition $((p, \overline{\Box}\varphi_1, c, a), b, Q)$ derived from the runs $I_1(p_j) \xrightarrow[A_1]{a_j} Q_j$ for all $1 \leq j \leq n$, with $Q = Q_1 \cup Q_n$. Suppose for some w, $w \in V^c_{\overline{\Box}\varphi_1}(q)$ for all $q \in Q$. By valuation soundness of A_1 we know $a_j w \in V^c_{\overline{\Box}\varphi_1}(I_1(p_j))$ and hence, since all transitions to $\langle p, abw\rangle$ are from configurations satisfying φ_1, we have $bw \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}\varphi_1, c, a)$ as required.

The remaining states are of the form $(p, \overline{\Box}\varphi_1, c)$. We first deal with the case when for all b we have $Pre(p, a, b) = \emptyset$. In this case, the valuation of $\overline{\Box}\varphi_1$ contains all words of the form aw for some w. Hence, all added transitions are trivially sound.

Otherwise, take a new transition $((p, \overline{\Box}\varphi_1, c), a, Q)$ derived from some b, the value of $\overline{Pop}(p) = \{(p_1, a_1), \ldots, (p_n, a_n)\}$ and for all $1 \leq j \leq n$, the runs $I_1(p_j) \xrightarrow[A_1]{w_j} Q'_j \xrightarrow[A_1]{b} Q^{pop}_j$, with $Q_{pop} = Q^{pop}_1 \cup Q^{pop}_n$, and the value of $\overline{Rew}(p, =) \{(p'_1, a'_1), \ldots, (p'_{n'}, a'_{n'})\}$ and for all $1 \leq j \leq n'$, the runs $I_1(p'_j) \xrightarrow[A_1]{a'_j} Q^{rew}_j$, with $Q_{rew} = Q^{rew}_1 \cup Q^{rew}_n$. Finally, $Q = \left\{(p, \overline{\Box}\varphi_1, c, a, b)\right\} \cup Q_{pop} \cup Q_{rew}$.

Suppose for some w, $w \in V^c_{\overline{\Box}\varphi_1}(q)$ for all $q \in Q_{pop}$. By valuation soundness of A_1 we know $a_j aw \in V^c_{\overline{\Box}\varphi_1}(I_1(p_j))$ and hence all pop transitions leading to $\langle p, aw\rangle$ are from configurations satisfying φ_1.

Now suppose for some aw, $aw \in V^c_{\overline{\Box}\varphi_1}(q)$ for all $q \in Q_{rew}$. By valuation soundness of A_1 we know $a_j w \in V^c_{\overline{\Box}\varphi_1}(I_1(p_j))$ and hence all rewrite transitions leading to $\langle p, aw\rangle$ are from configurations satisfying φ_1.

Finally, consider some bw in the valuation of $(p, \overline{\Box}\varphi_1, c, a)$. From the soundness of this state, shown above, we have that all push transitions leading to $\langle p, abw\rangle$ are from configurations satisfying φ_1.

Putting the three cases together, we have for all $abw \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}\varphi_1, c)$ as required.

The above cases do not cover the case $\perp \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}\varphi_1, c)$. However, since no push transition can reach this stack, we just require the first two cases and that $(p, \overline{\Box}\varphi_1, c, \perp) = q^{\varepsilon}_f$.

Case $BackDiamond(A, \varphi_1, c, \mathbb{P})$:

We assume that A is valuation sound with respect to some valuation V. By induction A_1 is valuation sound with respect to $V^c_{\varphi_1}$. We show that A' is valuation sound with respect to $V^c_{\overline{\Diamond}\varphi_1}$.

We begin with the states $(p, \overline{\Diamond}, c, a)$. Take a transition $((p, \overline{\Diamond}, c, a), b, Q)$. Then there is some $(p', a') \in \overline{Push}(p, a, b)$ such that $I_1(p') \xrightarrow{a'} QA_1$. From the soundness of A_1 we know for all w with $w \in V^c_{\overline{\Diamond}\varphi_1}(q)$ for all $q \in Q$ we have $a'w \in V^c\overline{\Diamond}\varphi_1(I_1(p'))$. Since $\langle p', a'w\rangle \hookrightarrow \langle p, abw\rangle$ we have $\langle p, abw\rangle$ satisfies φ_1 and hence $bw \in V^c_{\overline{\Diamond}\varphi_1}(p, \overline{\Diamond}, c, a)$ and the transition is sound.

For the remaining states, take a new transition $((p, \overline{\Diamond}\varphi_1, c), a, Q)$. There are three cases.

If the transition was derived from some $(p', a') \in \overline{Pop}(p)$ and the run $I_1(p') \xrightarrow[A_1]{a'a} Q$, then suppose for some w, $w \in V^c_{\overline{\Diamond}\varphi_1}(q)$ for all $q \in Q$. By valuation soundness of A_1 we know $a'aw \in V^c_{\overline{\Diamond}\varphi_1}(I_1(p'))$ and hence, since there is a transition

$\langle p', a'aw\rangle$, a configuration satisfying φ_1, to $\langle p, aw\rangle$ we obtain $aw \in V^c_{\overline{\Diamond}\varphi_1}(p, \overline{\Diamond}\varphi_1, c)$ as required.

If the transition was derived from some $(p', a') \in \overline{Rew}(p, a)$ and the run $I_1(p') \xrightarrow[A_1]{a'} Q$, then suppose for some w, $w \in V^c_{\overline{\Diamond}\varphi_1}(q)$ for all $q \in Q$. By valuation soundness of A_1 we know $a'w \in V^c_{\overline{\Diamond}\varphi_1}(I_1(p'))$ and hence, since there is a transition $\langle p', a'w\rangle$, a configuration satisfying φ_1, to $\langle p, aw\rangle$ we obtain $aw \in V^c_{\overline{\Diamond}\varphi_1}(p, \overline{\Diamond}\varphi_1, c)$ as required.

Finally, if $Q = \{(p, \overline{\Diamond}, c, a)\}$ then soundness is immediate from the definition of $V^c_{\overline{\Diamond}\varphi_1}$.

Case $LFP(A, Z, \varphi_1, c, \mathbb{P})$:

By assumption A is sound with respect to V. Let $V_\mu = V^c_{\mu Z.\varphi_1}$. Initially, A_0 is valuation sound with respect to V_μ since there are no transitions from the new states. Hence, we assume the case for A_i and prove it for A_{i+1}. By induction over the recursion, B_i is sound with respect to V_μ. Since I_i are sound with respect to φ_1 and (abusing notation) $\mu Z.\varphi_1 = \varphi_1(\mu Z.\varphi_1)$ we have that $B_i[Z/I_i]$ remains V_μ sound.

Take any transition $((p, \varphi, c), a, Q)$ or $((p, \varphi, c, b), a, Q)$ in A_{i+1} and any w such that for all $q \in Q$ we have $w \in V_\mu(q)$. Consider the corresponding transition $((p, \varphi, c+1), a, Q')$ or $((p, \varphi, c+1, b), a, Q')$ in $B_i[Z/I_i]$. All states q in Q' that are not level c or $c{+}1$ remain in Q, hence we have $w \in V_\mu(q)$. Furthermore, since the level c valuation of Z equals the level $c+1$ valuation, we have $w \in V_\mu(q)$ for all level c and $c+1$ states. Hence, by soundness of $B_i[Z/I_i]$ we know $aw \in V_\mu(p, \varphi, c+1)$ or $aw \in V_\mu(p, \varphi, c+1, b)$ and therefore $aw \in V_\mu(p, \varphi, c)$ or $aw \in V_\mu(p, \varphi, c, b)$ as required.

Case $GFP(A, Z, \varphi_1, c, \mathbb{P})$:

By assumption A is sound with respect to V. Let ν^α be $[\![\nu^\alpha Z.\varphi_1]\!]_{V_c}$. We begin, with a minor diversion.

Assume, A_i is valuation sound with respect to

$$V_{\alpha+1}(p, \varphi, c') = \begin{cases} V(p, \varphi, c') & \text{if } c' < c \\ \left\{ w \;\middle|\; \langle p, w\rangle \in \nu^{\alpha+1} \right\} & \text{if } \varphi = Z \text{ and } c = c' \\ \left\{ w \;\middle|\; \langle p, w\rangle \in [\![\varphi]\!]_{(V_c[Z \mapsto \nu^\alpha])} \right\} & \text{otherwise} \end{cases} .$$

and similarly for $V_{\alpha+1}(p, \varphi, c', a)$ (in the style of Definition 4.3). We show A_{i+1} is sound with respect to

$$V_{\alpha+2}(p, \varphi, c') = \begin{cases} V(p, \varphi, c') & \text{if } c' < c \\ \left\{ w \;\middle|\; \langle p, w\rangle \in \nu^{\alpha+2} \right\} & \text{if } \varphi = Z \text{ and } c = c' \\ \left\{ w \;\middle|\; \langle p, w\rangle \in [\![\varphi]\!]_{(V_c[Z \mapsto \nu^{\alpha+1}])} \right\} & \text{otherwise} \end{cases}$$

and similarly for $V_{\alpha+2}(p, \varphi, c', a)$. Let $V^{c+1}_{\alpha'} = (V_{\alpha'})^{c+1}_{\varphi_1}$. By induction, B_i is sound with respect to $V^{c+1}_{\alpha+1}$, which values Z as $\nu^{\alpha+1}$.

Take any $((p, Z, c), a, Q)$ in A_{i+1} and w such that $w \in V_{\alpha+2}(q)$. Take the corresponding transition $(I_i(p), a, Q')$ in B_i. For all $q \in Q'$ that are not level c or $c+1$ we know $q \in Q$ and hence $w \in V_{\alpha+2}(q)$ which is a subset of $V_{\alpha+1}(q)$. For level c states the same subset argument holds. For level $c+1$ the valuations are the same. Hence, the pre-conditions for the soundness condition are satisfied, and from the soundness of B_i we know $aw \in V^{c+1}_{\alpha+1}(I_i(p)) = \mu^{\alpha+2} = V_{\alpha+2}(p, Z, c)$, as required.

Take any $((p, \varphi, c), a, Q)$ or $((p, \varphi, c, b), a, Q)$ with $\varphi \neq Z$ in A_{i+1} and w such that $w \in V_{\alpha+2}(q)$. Take the corresponding transition $((p, \varphi, c+1), a, Q')$ or $((p, \varphi, c, b), a, Q')$ in B_i. For all $q \in Q'$ that are not level c or $c+1$ we know $q \in Q$ and hence $w \in V_{\alpha+2}(q)$ which is a subset of $V_{\alpha+1}(q)$. For level c states the same subset argument holds. For level $c+1$ the valuations are the same. Hence, the pre-conditions for the soundness condition are satisfied, and from the soundness of B_i we know $aw \in V^{c+1}_{\alpha+1}(p, \varphi, c+1) = V_{\alpha+2}(p, \varphi, c)$ or $aw \in V^{c+1}_{\alpha+1}(p, \varphi, c+1, b) = V_{\alpha+2}(p, \varphi, c, b)$, as required.

Thus, A_{i+1} is sound with respect to $V_{\alpha+2}$ as required.

We are now ready to prove the main result by induction over the ordinals. We have that, $A' = A_i = A_{i+1}$. A_0 is trivially sound with respect to V_0. Then, by the argument above, A_i is sound with respect to V_i. The case of a successor ordinal also follows from the above. For a limit ordinal λ, we have soundness for all $\alpha < \lambda$. Since Since $\theta^\lambda = \bigcap_{\alpha<\lambda} \theta^\alpha$, the result follows because each configuration in the limit appears in all smaller approximants, and we are sound for all smaller approximants (and trivially for the zeroth approximant). To regain the induction hypothesis for successor ordinals, we simply apply the successor construction once, which keeps all (p, φ, c) where $\varphi \neq Z$ sound for the limit, while (p, Z, c) becomes sound for $\nu^{\lambda+1}$. $\square$

Lemma 4.4 (Valuation Completeness). *The algorithm is V-complete.*

Proof. The proof is by induction over the recursion. The base cases $\widehat{x}$ and Z are immediate.

Case $And(A, \varphi_1, \varphi_2, c, \mathbb{P})$:

By assumption, A is valuation complete with respect to some V. Furthermore, by induction, A_1 and A_2 are valuation complete with respect to $V^c_{\varphi_1}$ and $V^c_{\varphi_2}$ respectively. We have $V^c_{\varphi_1 \wedge \varphi_2}$ as above. We claim A' is complete with respect to this valuation. This only has to be shown for the new states of the form $q_{new} = (p, \varphi_1 \wedge \varphi_2, c)$. Suppose $aw \in V^c_{\varphi_1 \wedge \varphi_2}(q_{new})$. This implies $aw \in V^c_{\varphi_1}(I_1(p))$ and $aw \in V^c_{\varphi_2}(I_2(p))$. Since A_1 and A_2 are valuation complete, we have some transitions $(I_1(p), a, Q_1)$ and $(I_2(p), a, Q_2)$ such that for all $q \in Q_1 \cup Q_2$, $w \in V^c_{\varphi_1 \wedge \varphi_2}(q)$. This implies the transition $((p, \varphi_1 \wedge \varphi_2, c), a, Q_1 \cup Q_2)$ is in A'. This transition witnesses completeness.

Case $Or(A, \varphi_1, \varphi_2, c, \mathbb{P})$:

By assumption, A is valuation complete with respect to some V. Furthermore, by induction, A_1 and A_2 are valuation complete with respect to $V^c_{\varphi_1}$ and $V^c_{\varphi_2}$

respectively. Take $V^c_{\varphi_1 \vee \varphi_2}$ as above. We claim A' is complete with respect to this valuation. This only has to be shown for the new states of the form $q_{new} = (p, \varphi_1 \vee \varphi_2, c)$. Suppose $aw \in V^c_{\varphi_1 \vee \varphi_2}(q_{new})$. This implies $aw \in V^c_{\varphi_1}(I_1(p))$ or $aw \in V^c_{\varphi_2}(I_2(p))$. We assume the first case by symmetry. Since A_1 is valuation complete, we have some transition $(I_1(p), a, Q)$ such that for all $q \in Q$, $w \in V^c_{\varphi_1}(q)$. This implies the transition $((p, \varphi_1 \vee \varphi_2, c), a, Q)$ is in A'. This transition witnesses completeness.

Case $Box(A, \varphi_1, c, \mathbb{P})$:

We are given that A is valuation complete with respect to some valuation V, and by induction we have completeness of A_1 with respect to $V^c_{\varphi_1}$. We show A' is complete with respect to $V^c_{\Box\varphi_1}$.

In the case that $Next(p, a) = \emptyset$, we either have $a = \perp$ and the transition from $(p, \Box\varphi_1, c)$ to $\left\{q^\varepsilon_f\right\}$ witnesses completeness, or we have $a \neq \perp$ and the transition from $(p, \Box\varphi_1, c)$ to $\{q^*\}$ witnesses completeness.

Otherwise, assume we have aw such that $aw \in V^c_{\Box\varphi_1}(p, \Box\varphi_1, c)$ and $Next(p, a) = \{(p_1, w_1), \ldots, (p_n, w_n)\}$. Hence, for all $1 \leq j \leq n$, we have $w_j w \in V^c_{\Box\varphi_1}(I_1(p_j))$. By completeness of A_1 we have runs $I_1(p_j) \xrightarrow[A_1]{w_j} Q_j$ such that for all $q \in Q_j$, $w \in V^c_{\Box\varphi_1}(q)$. Hence, the transition $((p, \Box\varphi_1, c), a, Q_1 \cup \cdots \cup Q_n)$ witnesses completeness.

Case $Diamond(A, \varphi_1, c, \mathbb{P})$:

We are given that A is valuation complete with respect to some valuation V, and by induction we have completeness of A_1 with respect to $V^c_{\varphi_1}$. We show A' is complete with respect to $V^c_{\Diamond\varphi_1}$.

Assume some aw such that $aw \in V^c_{\Diamond\varphi_1}(p, \Diamond\varphi_1, c)$ and take $(p', w') \in Next(p, a)$ such that we have $\langle p', w'w\rangle \in V^c_{\Diamond\varphi_1}(I_1(p'))$. By completeness of A_1 we have a run $I_1(p') \xrightarrow[A_1]{w'} Q$ such that for all $q \in Q$, $w \in V^c_{\Diamond\varphi_1}(q)$. Hence, the transition $((p, \Diamond\varphi_1, c), a, Q)$ witnesses completeness.

Case $BackBox(A, \varphi_1, c, \mathbb{P})$:

We are given that A is valuation complete with respect to some valuation V, and by induction we have completeness of A_1 with respect to $V^c_{\varphi_1}$. We show A' is complete with respect to $V^c_{\overline{\Box}\varphi_1}$.

As in the soundness proof, we begin with the states $(p, \overline{\Box}\varphi_1, c, a)$. In the case $\overline{Push}(p, a, b) = \emptyset$ for some b, we either have $b = \perp$ and the transition from $(p, \Box\varphi_1, c, a)$ to $\left\{q^\varepsilon_f\right\}$ witnesses completeness, or we have $a \neq \perp$ and the transition to $\{q^*\}$ witnesses completeness.

Otherwise $\overline{Push}(p, a, b) = \{(p_1, a_1), \ldots, (p_n, a_n)\}$. Take some bw such that $abw \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}\varphi_1, c, a)$. Then we have $a_j w \in V^c_{\overline{\Box}\varphi_1}(p_j, \varphi_1, c)$ for all $1 \leq j \leq n$. From completeness of A_1 we have a transition $I_1(p_j) \xrightarrow{a_j} Q_j$ with $w \in V^c_{\overline{\Box}\varphi_1}(q)$

for all $q \in Q_j$. Hence, we have a complete b-transition from $(p, \overline{\Box}\varphi_1, c, a)$ as required.

For the states of the form $(p, \overline{\Box}\varphi_1, c)$ we first deal with the case when for all b we have $Pre(p, a, b) = \emptyset$. In this case we immediately have transitions witnessing completeness.

Otherwise, take some $abw \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}\varphi_1, c)$. Then, for all $(p', a') \in \overline{Pop}(p)$, we have $a'abw \in V^c_{\overline{\Box}\varphi_1}(I_1(p'))$; and for all $(p', a') \in \overline{Rew}(p, a)$ we have $a'bw \in V^c_{\overline{\Box}\varphi_1}(I_1(p'))$; and for all $(p', a') \in \overline{Push}(p, a, b)$ we have $a'w \in V^c_{\overline{\Box}\varphi_1} I_1(p')$. From completeness of A_1 we have a complete run $I_1(p') \xrightarrow[A_1]{a'} Q' \xrightarrow[A_1]{a} Q$ for each $(p', a') \in \overline{Pop}(p)$ and a complete run $I_1(p') \xrightarrow[A_1]{a'} Q$ for each $(p', a') \in \overline{Rew}(p, a)$. Since we know $bw \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}\varphi_1, c, a)$ there must be some complete transition from $(p, \overline{\Box}\varphi_1, c)$ as required.

The only case not covered by the above is the case $\perp \in V^c_{\overline{\Box}\varphi_1}(p, \overline{\Box}, \varphi_1, c)$. In this case there are no push transitions reaching this configuration. That is $\overline{Push}(p, \perp, b) = \emptyset$ for all b. Note also that we equated all $(p, \overline{\Box}\varphi_1, c, \perp)$ with q^ε_f. Hence, from the pop and rewrite cases above, and that $(p, \overline{\Box}\varphi_1, c, \perp) = q^\varepsilon_f$ we have completeness as required.

Case $BackDiamond(A, \varphi_1, c, \mathbb{P})$:

We are given that A is valuation complete with respect to some valuation V, and by induction we have completeness of A_1 with respect to $V^c_{\varphi_1}$. We show A' is complete with respect to $V^c_{\overline{\Diamond}\varphi_1}$. There are three cases.

Assume some aw such that $aw \in V^c_{\overline{\Diamond}\varphi_1}(p, \overline{\Diamond}\varphi_1, c)$ by virtue of some $(p', a') \in \overline{Pop}(p)$ such that we have $\langle p', a'aw\rangle \in V^c_{\overline{\Diamond}\varphi_1}(I_1(p'))$. By completeness of A_1 we have a run $I_1(p') \xrightarrow[A_1]{a'a} Q$ such hat for all $q \in Q$, $w \in V^c_{\overline{\Diamond}\varphi_1}(q)$. Hence, the transition $((p, \overline{\Diamond}\varphi_1, c), a, Q)$ witnesses completeness.

Otherwise, take some aw such that $aw \in V^c_{\overline{\Diamond}\varphi_1}(p, \overline{\Diamond}\varphi_1, c)$ from some $(p', a') \in \overline{Rew}(p, a)$ such that we have $\langle p', a'w\rangle \in V^c_{\overline{\Diamond}\varphi_1}(I_1(p'))$. By completeness of A_1 we have a run $I_1(p') \xrightarrow[A_1]{a'} Q$ such that for all $q \in Q$, $w \in V^c_{\overline{\Diamond}\varphi_1}(q)$. Hence, the transition $((p, \overline{\Diamond}\varphi_1, c), a, Q)$ witnesses completeness.

Finally, take some abw such that $abw \in V^c_{\overline{\Diamond}\varphi_1}(p, \overline{\Diamond}\varphi_1, c)$ from some $(p', a') \in \overline{Push}(p, a, b)$ such that we have $\langle p', a'w\rangle \in V^c_{\overline{\Diamond}\varphi_1}(I_1(p'))$. By completeness of A_1 we have a run $I_1(p') \xrightarrow[A_1]{a'} Q$ such that for all $q \in Q$, $w \in V^c_{\overline{\Diamond}\varphi_1}(q)$. Hence, the transitions $((p, \overline{\Diamond}\varphi_1, c), a, \{(p, \overline{\Diamond}, c, a)\})$ and $((p, \overline{\Diamond}\varphi_1, c, a), a, Q)$ witness completeness.

Case $LFP(A, Z, \varphi_1, c, \mathbb{P})$:

By assumption A is complete with respect to V. Let μ^α be $[\![\mu^\alpha Z.\varphi_1]\!]_{V_c}$. We begin, as before, with a minor diversion.

Assume, A_i is valuation complete with respect to

$$V_{\alpha+1}(p, \varphi, c') = \begin{cases} V(p, \varphi, c') & \text{if } c' < c \\ \left\{ w \;\middle|\; \langle p, w\rangle \in \mu^{\alpha+1} \right\} & \text{if } \varphi = Z \text{ and } c = c' \\ \left\{ w \;\middle|\; \langle p, w\rangle \in [\![\varphi]\!]_{(V_c[Z\mapsto\mu^\alpha])} \right\} & \text{otherwise} \end{cases}$$

and similarly for $V_{\alpha+1}(p, \varphi, c', a)$ (in the spirit of Definition 4.3). We show A_{i+2} is complete with respect to

$$V_{\alpha+2}(p, \varphi, c') = \begin{cases} V(p, \varphi, c') & \text{if } c' < c \\ \left\{ w \;\middle|\; \langle p, w\rangle \in \mu^{\alpha+2} \right\} & \text{if } \varphi = Z \text{ and } c = c' \\ \left\{ w \;\middle|\; \langle p, w\rangle \in [\![\varphi]\!]_{(V_c[Z\mapsto\mu^{\alpha+1}])} \right\} & \text{otherwise} \end{cases}$$

and similarly for $V_{\alpha+1}(p, \varphi, c', a)$. Let $V_{\alpha'}^{c+1} = (V_{\alpha'})_{\varphi_1}^{c+1}$. By induction, B_i is complete with respect to $V_{\alpha+1}^{c+1}$, which values Z as $\mu^{\alpha+1}$.

For each (p, Z, c) in A_{i+1}, take some $aw \in V_{\alpha+2}(p, Z, c) = \mu^{\alpha+2}$. Since $\mu^{\alpha+2} = \varphi_1(\mu^{\alpha+1})$ we have that $aw \in V_{\alpha+1}^{c+1}(I_i(p))$ from the completeness of B_i. Hence there was a complete transition $(I_i(p), a, Q)$ in B_i. For all states $q \in Q$ not of level c or $c + 1$, the completeness conditions remain satisfied after the projections in A_{i+1}. For level c state (p', φ, c) we know that $w \in V_{\alpha+1}(p', \varphi, c)$ which is a subset of $V_{\alpha+2}(p', \varphi, c)$ and we are done. For a level $c + 1$ state $(p', \varphi, c{+}1)$ we know $w \in V_{\alpha+1}^{c+1}(p', \varphi, c{+}1)$ which is also a subset of $V_{\alpha+2}(p', \varphi, c)$, hence we are done.

For each (p, φ, c) or (p, φ, c, a) in A_{i+1} with $\varphi \neq Z$, take some $aw \in V_{\alpha+2}(p, \varphi, c)$ or $aw \in V_{\alpha+2}(p, \varphi, c, a)$. From the completeness of B_i there was a complete transition $((p, \varphi, c + 1), a, Q)$ or $((p, \varphi, c + 1, a), aQ)$ in B_i. For all states $q \in Q$ not of level c or $c + 1$, the completeness conditions remain satisfied after the projections in A_{i+1}. For a level c state (p', φ', c) or (p', φ', c, a') we know that $w \in V_{\alpha+1}(p', \varphi', c)$ which is a subset of $V_{\alpha+2}(p', \varphi', c)$ or $w \in V_{\alpha+1}(p', \varphi', c, a')$ which is a subset of $V_{\alpha+2}(p', \varphi', c, a')$ and we are done. For a level $c + 1$ state $(p', \varphi', c + 1)$ or $(p', \varphi', c + 1, a')$ we know $w \in V_{\alpha+1}^{c+1}(p', \varphi', c + 1)$ which is also a subset of $V_{\alpha+2}(p', \varphi', c)$ or $w \in V_{\alpha+1}^{c+1}(p', \varphi', c + 1, a)$ which is also a subset of $V_{\alpha+2}(p', \varphi', c, a)$, hence we are done.

Thus, A_{i+1} is complete with respect to $V_{\alpha+2}$ as required.

We are now ready to prove the main result by induction over the ordinals. Trivially, $A' = A_i = A_{i+1}$ (for some $i \geq 1$) is sound with respect to V_0. This is because A_0 is complete with respect to the extension of V mapping Z to μ^0, and the recursive call ensures completeness with respect to the full V_0. The case of a successor ordinal was shown above. For a limit ordinal λ, we have completeness for V_α for all $\alpha < \lambda$. Since $\mu^\lambda = \bigcup_{\alpha<\lambda} \mu^\alpha$, the result follows because each configuration in the limit appears in some smaller approximant, and the transition witnessing completeness for the approximant witnesses completeness for the limit. To regain the induction hypothesis for successor ordinals, we simply apply the successor construction once, which keeps all (p, φ, c) where $\varphi \neq Z$ complete for the limit, while (p, Z, c) becomes complete for $\mu^{\lambda+1}$.

Case $GFP(A, Z, \varphi_1, c, \mathbb{P})$:

By assumption A is complete with respect to V. Initially, A_0 is valuation complete with respect to the extension of V that values Z as $[\![\nu Z.\varphi_1]\!]_{V_c}$. After the first iteration, using a specialisation of the argument below, we have that A_1 is complete with respect to $V^c_{\nu Z.\varphi_1}$, which we will abbreviate as V_ν.

We assume completeness with respect to V_ν for A_i and prove it for A_{i+1}. By induction over the recursion, B_i is complete with respect to V_ν. Since I_i are complete and $\nu Z.\varphi_1 = \varphi_1(\nu Z.\varphi_1)$ we have that $B_i[Z/I_i]$ remains V_ν complete.

Take any $aw \in V_\nu(p, \varphi, c)$ or $aw \in V_\nu(p, \varphi, c, b)$. Since $V_\nu(p, \varphi, c) = V_\nu(p, \varphi, c+1)$ and $V_\nu(p, \varphi, c, b) = V_\nu(p, \varphi, c+1, b)$ we have a transition $((p, \varphi, c+1), a, Q)$ or $((p, \varphi, c+1, b), a, Q)$ in $B_i[Z/I_i]$ that witnesses completeness for $B_i[Z/I_i]$. From this transition we have $((p, \varphi, c), a, \pi_c(Q))$ or $((p, \varphi, c, b), a, \pi_c(Q))$ in A_{i+1}. For all $q \in Q$ of level less than c we have from B_i that $w \in V_\nu(q)$. For q of level c and $c+1$ we have $w \in V_\nu(\pi_c(q))$ from $V_\nu(p', \varphi', c) = V_\nu(p', \varphi', c+1)$ and $V_\nu(p', \varphi', c, b') = V_\nu(p', \varphi', c+1, b')$ for all p', b' and φ'. Hence we have a transition witnessing completeness, as required. □

5. Termination and Correctness of *Denotation*$(\chi, A_V, \mathbb{P})$

Termination and valuation soundness and completeness for the called subroutines are given in Lemma 3.4, Lemma 4.3 and Lemma 4.4.

Theorem 5.1. *Let* $(A, I) = \mathit{Denotation}(\chi, A_V, \mathbb{P})$ *where* A_V *describes a valuation* V*. The states* I *of* A *give the denotation* $[\![\chi]\!]^{\mathbb{P}}_V$*.*

Proof. Observe that A_V is automatically V-sound and -complete. There are two cases when χ is not $\widehat{x}$ or Z. Either $I = \{\ (p, Z, 1) \mid p \in \mathcal{P}\ \}$ when $\chi = \sigma Z.\varphi(Z)$ for $\sigma \in \{\mu, \nu\}$, or $I = \{\ (p, \chi, 1) \mid p \in \mathcal{P}\ \}$ otherwise. In both cases, from Lemma 4.3 with Lemma 4.1 and Lemma 4.4 with Lemma 4.2 we have the theorem as required. □

[1] A. Arnold and D. Niwiński. *Rudiments of μ-Calculus.* Elsevier, Amsterdam, The Netherlands, 2001.

[2] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *International Conference on Concurrency Theory*, pages 135–150, 1997.

[3] A. Finkel, B. Willems, and P. Wolper. A direct symbolic approach to model checking pushdown systems. In *Proc. 2nd Int. Workshop on Verification of Infinite State Systems (INFINITY'97), Bologna, Italy, July 11–12, 1997*, volume 9 of *Electronic Notes in Theor. Comp. Sci.* Elsevier, 1997.

[4] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *SFCS '91: Proceedings of the 32nd annual symposium on Foundations of computer science*, pages 368–377, Washington, DC, USA, 1991. IEEE Computer Society.

[5] M. Hague and C.-H. L. Ong. Winning regions of pushdown parity games: A saturation method. In *Concurrency Theory, Proceedings of 20th International Conference, CONCUR 2009, Bologna, Italy, September 1-4, 2009*, volume 5710, pages 384–398. Springer-Verlag, 2009. LNCS.

[6] I. Walukiewicz. Pushdown processes: Games and model checking. In Rajeev Alur and Thomas A. Henzinger, editors, *Proceedings of the Eighth International Conference on Computer Aided Verification CAV*, volume 1102, pages 62–74, New Brunswick, NJ, USA, / 1996. Springer Verlag.

[7] J. C. Bradfield and C. P. Stirling. Modal logics and mu-calculi: An introduction. In *Handbook of Process Algebra*, pages 293–330, 2001.

[8] J. Esparza, A. Kučera, and S. Schwoon. Model-checking LTL with regular valuations for pushdown systems. In *Proc. of TACS 2001*, number 2215 in Lecture Notes in Computer Science, pages 306–339, 2001.

[9] K. Etessami. Analysis of recursive game graphs using data flow equations. In *VMCAI*, pages 282–296, 2004.

[10] N. D. Jones and S. S. Muchnick. Even simple programs are hard to analyze. In *POPL '75: Proceedings of the 2nd ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 106–118, New York, NY, USA, 1975. ACM.

[11] N. Piterman and M. Y. Vardi. Global model-checking of infinite-state systems. In *CAV*, pages 387–400, 2004.

[12] O. Burkart and B. Steffen. Composition, decomposition and model checking of pushdown processes. *Nordic J. of Computing*, 2(2):89–125, 1995.

[13] O. Burkart and B. Steffen. Model checking the full modal mu-calculus for infinite sequential processes. *Theor. Comput. Sci.*, 221(1-2):251–270, 1999.

[14] O. Serre. Note on winning positions on pushdown games with ω-regular conditions. *Information Processing Letters*, 85:285–291, 2003.

[15] R. V. Book and F. Otto. *String-Rewriting Systems*. Springer-Verlag, 1993.

[16] S. Schwoon. *Model-checking Pushdown Systems*. PhD thesis, Technical University of Munich, 2002.

[17] T. Ball and S. K. Rajamani. Bebop: A symbolic model checker for boolean programs. In *Proceedings of the 7th International SPIN Workshop on SPIN Model Checking and Software Verification*, pages 113–130, London, UK, 2000. Springer-Verlag.

[18] T. Ball and S. K. Rajamani. The SLAM project: Debugging system software via static analysis. In *Conference Record of POPL'02: The 29th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 1–3, Portland, Oregon, January 16–18, 2002.

[19] T. Cachat. Symbolic strategy synthesis for games on pushdown graphs. In *ICALP '02: Proceedings of the 29th International Colloquium on Automata, Languages and Programming*, pages 704–715, London, UK, 2002. Springer-Verlag.

[20] T. Cachat. *Games on Pushdown Graphs and Extensions.* PhD thesis, RWTH Aachen, 2003.

[21] T. Reps, S. Schwoon, S. Jha, and D. Melski. Weighted pushdown systems and their application to interprocedural dataflow analysis. *Sci. Comput. Program.*, 58(1-2):206–263, 2005.