Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Global Model Checking on Pushdown Multi-Agent Systems*

Taolue Chen
Department of Computer Science
Middlesex University
London, United Kingdom
t.chen@mdx.ac.uk

Fu Song
Shanghai Key Laboratory of Trustworthy Computing
East China Normal University
Shanghai, P.R.China
fsong@sei.ecnu.edu.cn

Zhilin Wu
State Key Laboratory of Computer Science
Institute of Software
Chinese Academy of Sciences
Beijing, P.R.China
wuzl@ios.ac.cn

Abstract

Pushdown multi-agent systems, modeled by *pushdown* game structures (PGSs), are an important paradigm of *infinite-state* multi-agent systems. Alternating-time temporal logics are well-known specification formalisms for multi-agent systems, where the selective path quantifier is introduced to reason about strategies of agents. In this paper, we investigate model checking algorithms for variants of alternating-time temporal logics over PGSs, initiated by Murano and Perelli at IJCAI'15. We first give a triply exponential-time model checking algorithm for ATL* over PGSs. The algorithm is based on the saturation method, and is the first *global* model checking algorithm with a matching lower bound. Next, we study the model checking problem for the alternating-time μ-calculus. We propose an exponential-time global model checking algorithm which extends similar algorithms for pushdown systems and modal μ-calculus. The algorithm admits a matching lower bound, which holds even for the alternation-free fragment and ATL.

1 Introduction

Over the last two decades, *model checking* has become an attractive approach for verifying correctness of systems. Given a model of a system, model checking exhaustively and automatically checks whether this model meets a given specification. It is widely used to verify protocol, hardware design and software (Baier and Katoen 2008). Classical model checking usually focuses on (finite) Kripke structures against properties specified in logical formulae such as Linear Temporal Logic (LTL) (Pnueli 1977) and Computational Tree Logic (CTL) (Clarke and Emerson 1981).

Model checking has been extended to *multi-agent systems* which have been successfully employed as a modeling paradigm in a number of scenarios such as autonomous spacecraft control (Muscettola et al. 1998). A multi-agent system is a complex decentralized computing system composed of multiple interacting intelligent agents within an environment, in which the behavior of each agent is determined by its observed information of the system. To specify the behavior of multi-agent systems, a well-known logical formalism is Alternating-Time Temporal Logics (Alur, Henzinger, and Kupferman 2002) for which a model checking algorithm for *finite-state* concurrent game structures is also given therein. Model checking algorithms for various other temporal logics on *finite* multi-agent systems have been proposed in several works, e.g. (Bourahla and Benmohamed 2005; Bulling and Jamroga 2011; Jamroga and Murano 2015).

More recently, on a different dimension, model checking for ATL* on a class of *infinite-state* multi-agent systems, i.e., *pushdown multi-agent systems*, was also studied in (Murano and Perelli 2015). The authors introduced *pushdown game structures* (PGSs) as the model, showed that the model checking problem is 2EXPSPACE-hard, and proposed a model checking algorithm in 3EXPTIME. Our work follows the direction of (Murano and Perelli 2015).

We note that in the literature there is a distinction between *local* and *global* model checking. In the former setting one is given a specific state of the system and determines whether it satisfies a given property. In the latter setting one computes (a finite representation of) the set of states that satisfy a given property. The importance of global model checking has been discussed in (Piterman and Vardi 2004). In a nutshell, it is crucial when repeated checks are required, or where the model checking is only a component of the verification process. As a matter of fact, for many years global model checking algorithms were the standard. Moreover, obviously one can reduce local model checking to the global counterpart, but not vice versa when an infinite state space is concerned, as in the current setting.

The algorithm of (Murano and Perelli 2015), which is based on (a variant of) tree automata, is *local*. (Technically the algorithm works on the product of PGSs and tree automata in a *top-down* fashion, and it is open whether this can be done in a *bottom-up* way, as mentioned by the authors.) In contrast, we investigate the *global* model checking problem for alternating-time temporal logics on PGSs. Namely we aim to compute the set of all of the configurations satisfying the formula in some alternating-time temporal logics. In this work, we consider two variants of alternating-time temporal logics: ATL* and alternating-time μ-calculus.

Concerning the global model checking for ATL* on PGSs, as one of the main contributions, we present a re-

*Authors are alphabetically ordered.

duction from the problem to checking non-emptiness of alternating parity pushdown systems which can be solved using approaches given in (Hague and Ong 2009). Our global model checking approach has the same complexity upper bound as the local model checking algorithm proposed in (Murano and Perelli 2015). We also show that the model checking problem for ATL* is 3EXPTIME-complete which improves the 2EXPSPACE lower bound of (Murano and Perelli 2015). One of the features of our algorithm is that it can deal with *regular valuations* rather than *simple valuations* of atomic propositions. By regular valuations one atomic proposition can denote an infinite (but regular) set of configurations. This turns out be a very handy specification approach (see examples in (Esparza, Kucera, and Schwoon 2003)). As we use alternating multi-automata as the "data structure" for configurations of PGSs, this comes almost for free, while it is unclear to us how the algorithm of (Murano and Perelli 2015) can support this in an immediate way.

In verification, modal μ-calculus is generally considered to be the "assembly language" of various specifications in the sense that most temporal logics can be translated into μ-calculus to obtain a uniform model checking algorithm. In the multi-agent system setting, an alternating-time extension of μ-calculus has been considered in the original paper (Alur, Henzinger, and Kupferman 2002) already. As another contribution, we study model checking algorithms for alternating-time μ-calculus (AMC) over PGSs. We extend the saturation method of (Hague and Ong 2011) for modal μ-calculus over pushdown systems to the multi-agent system setting, obtaining an EXPTIME-time algorithm. Thanks to the alternating multi-automata as the data structure again, we can cope with the selective path quantifier of the alternating-time logic directly while keeping constructions for other μ-calculus operators untouched. The algorithm inherits the advantages of its counterpart in (Hague and Ong 2011), i.e., simple, amenable to implementation (in practice it is *the* implementation technique for pushdown model checkers), and efficient. For the lower bound, we show that the global model checking problem for *alternation-free* AMC on PGSs is already EXPTIME-hard. In fact, we prove that a simple formula $\langle\langle A\rangle\rangle \mathbf{F} q$ (which is equivalent to $\mu Z.\ \langle\langle A\rangle\rangle \mathbf{X} Z \vee q$) is sufficient to obtain the EXPTIME hardness. From this, we also deduce that model checking of ATL (the alternating-time counterpart of CTL) on PGSs is EXPTIME-complete. To the best of our knowledge, this result is also new.

We remark that it is known that AMC is more expressive than ATL* (Alur, Henzinger, and Kupferman 2002). However, this does *not* contradict the complexity bounds we obtained here, as translating an ATL* formula into an equivalent AMC formula involves a doubly exponential blow-up in the size of the formula (de Alfaro, Henzinger, and Majumdar 2001; Alur, Henzinger, and Kupferman 2002), and model checking AMC over PGSs is exponential with respect to sizes of formulae.

2 Preliminaries

We fix the following notations. Let AP be a finite set of atomic propositions, Ag be a finite set of agents, Ac be a finite set of actions that can be made by agents, $Dc = Ac^{Ag}$ be the set of *decisions* of the agents in Ag. For each agent $a \in Ag$ and decision $d \in Dc$, let $d(a)$ denote the action made by the agent a in d.

Given a set X, an X-labeled tree is a pair (T_r, r), where T_r is a prefix closed subset of $\mathbb{N}^*$, and $r : T_r \to X$ is a labeling function that assigns to each node an element from X. The root of a tree is ϵ, and for each node $t \in T_r$, if there is $i \in \mathbb{N}$ s.t. $ti \in T_r$, then ti is called a *child* of t, otherwise, t is called a *leaf*. A *path* π of (T_r, r) is a least subset of T_r such that $\epsilon \in \pi$, and for every $t \in \pi$, either t is a leaf in (T_r, r) or there is exactly one $i \in \mathbb{N}$ s.t. $ti \in \pi$. Given a path π, let $r(\pi) = x_0 x_1 ...$ denote the sequence of labeled elements of the path π (in the order of the lengths $|t|$ of the nodes $t \in \pi$).

Pushdown Game Structures

Definition 1. *(Murano and Perelli 2015) A Pushdown Game Structure (PGS) is a tuple $\mathcal{P} = (P, \Gamma, \Delta, \lambda)$, where P is a finite set of control states, Γ is a finite stack alphabet, $\Delta : P \times \Gamma \times Dc \to P \times \Gamma^*$ is a transition function, $\lambda : P \times \Gamma^* \to 2^{AP}$ is a labeling function that assigns to each $\langle p, \omega\rangle \in P \times \Gamma^*$ a set of atomic propositions. W.l.o.g., we assume that $\perp \in \Gamma$ is a special bottom stack symbol never popped up from the stack.*

A *configuration* of the PGS $\mathcal{P}$ is a pair $\langle p, \omega\rangle$, where $p \in P$ is the control state, $\omega \in \Gamma^*$ is the stack content. Let $\mathcal{C}_\mathcal{P}$ denote the set $P \times \Gamma^*$ of all the configurations of the PGS $\mathcal{P}$. For every $(p, \gamma, d) \in P \times \Gamma \times Dc$ such that $\Delta(\langle p, \gamma\rangle, d) = (p', \omega)$, we sometimes write $\langle p, \gamma\rangle \overset{d}{\hookrightarrow}_\mathcal{P} \langle p', \omega\rangle$ instead.

The transition relation $\Longrightarrow_\mathcal{P}$: $\mathcal{C}_\mathcal{P} \times Dc \times \mathcal{C}_\mathcal{P}$ of the PGS $\mathcal{P}$ is defined as follows: for every $\omega' \in \Gamma^*$, if $\langle p, \gamma\rangle \overset{d}{\hookrightarrow}_\mathcal{P} \langle p', \omega\rangle$, then $\langle p, \gamma\omega'\rangle \overset{d}{\Longrightarrow}_\mathcal{P} \langle p', \omega\omega'\rangle$. Given a pair $\langle p, \gamma\rangle \in P \times \Gamma$ and a function $f : A \to Ac$ such that $A \subseteq Ag$, let $succ_f(p, \gamma)$ denote the set of tuples $\{\langle p', \omega\rangle \mid \langle p, \gamma\rangle \overset{d}{\hookrightarrow}_\mathcal{P} \langle p', \omega\rangle \in \Delta \text{ and } \forall a \in A : d(a) = f(a)\}$, and $succ_f(p, \gamma\omega')$ denote the set of configurations $\{\langle p', \omega\omega'\rangle \mid \langle p', \omega\rangle \in succ_f(p, \gamma)\}$ for every $\omega' \in \Gamma^*$ which is the set of all the possible successors of $\langle p, \gamma\omega'\rangle$ on the actions $f(a)$ for $a \in A$ (agents $Ag \setminus A$ can make any action).

A *track* in a PGS $\mathcal{P}$ is a *finite* sequence π of configurations $c_0 ... c_n$ such that for every $i : 0 \le i < n$, $c_i \overset{d}{\Longrightarrow}_\mathcal{P} c_{i+1}$. A *path* in a PGS $\mathcal{P}$ is an *infinite* sequence π of configurations $c_0 c_1 ...$ such that for every $i \ge 0$, $c_i \overset{d}{\Longrightarrow}_\mathcal{P} c_{i+1}$. Given a track $\pi = c_0 ... c_n$ (resp. path $\pi = c_0 c_1 ...$), for every $i : 0 \le i \le n$ (resp. $i \ge 0$), let π_i denote the configuration c_i, $\pi_{\ge i}$ denote the suffix sequence $c_i ... c_n$ (resp. $c_i c_{i+1} ...$) of π, $\pi_{<i}$ denote the prefix sequence $c_0 ... c_{i-1}$ of π. Let $T_\mathcal{P} \subseteq \mathcal{C}_\mathcal{P}^+$ denote the set of all the tracks in $\mathcal{P}$, $\prod_\mathcal{P} \subseteq \mathcal{C}_\mathcal{P}^\omega$ denote the set of all the paths in $\mathcal{P}$. Given a configuration c, let $T_\mathcal{P}(c) = \{\pi \in T_\mathcal{P} \mid \pi_0 = c\}$ denote the set of all the tracks starting from c, $\prod_\mathcal{P}(c) = \{\pi \in \prod_\mathcal{P} \mid \pi_0 = c\}$ denote the set of all the paths starting from c.

A *strategy* for an agent in a PGS $\mathcal{P}$ is a function $\theta : T_\mathcal{P} \to Ac$ that contains all the possible choices of actions depending upon the tracks (i.e., the history the agent saw so far). Let Θ denote the set of all the possible strategies. Given a set of

agents $A \subseteq Ag$, an *assignment* over A is a function $\upsilon_A : A \to \Theta$ that assigns to each agent a strategy. Let V denote the set of all the possible assignments. A path π is *compatible* with an assignment υ_A over the set A of agents, if for every $i \geq 0$, there is a decision $d \in Dc$ such that $\pi_i \stackrel{d}{\Longrightarrow}_{\mathcal{P}} \pi_{i+1}$ and $d(a) = \upsilon_A(a)(\pi_{<i})$ for all $a \in A$. Given a configuration $c \in \mathcal{C}_{\mathcal{P}}$ and an assignment υ_A over the set A of agents, let $\prod_{\mathcal{P}}(c, \upsilon_A)$ denote the set of paths starting from c that are compatible with respect to the assignment υ_A. Formally, $\prod_{\mathcal{P}}(c, \upsilon_A) = \{\pi \mid \pi \in \prod_{\mathcal{P}}(c) \wedge \pi \text{ is compatible with } \upsilon_A\}$.

Alternating-Time Temporal Logics

In this subsection, we recall the definitions of two alternating-time temporal logics: ATL* and alternating-time μ-calculus (AMC). ATL* and AMC were proposed by Alur et al. in (Alur, Henzinger, and Kupferman 2002) as extensions of CTL* and modal μ-calculus (Kozen 1983), where the universal and existential path quantifiers are replaced by more general quantifiers, called path selective quantifiers, each of which is parameterized by a set of agents. Alternating-time temporal logics are defined with respect to a set of atomic propositions AP and a set of agents Ag.

Definition 2. *ATL* formulae are defined by the following grammar:*

$$\phi ::= q \mid \neg\phi \mid \phi \wedge \phi \mid \boldsymbol{X}\phi \mid \phi \boldsymbol{U} \phi \mid \langle\langle A\rangle\rangle\phi$$

where $q \in AP$, $A \subseteq Ag$.

We let $\phi_1 \vee \phi_2 \stackrel{\triangle}{=} \neg(\neg\phi_1 \wedge \neg\phi_2)$, $\mathbf{F}\phi \stackrel{\triangle}{=} true\ \mathbf{U}\ \phi$, $\mathbf{G}\phi \stackrel{\triangle}{=} \neg\mathbf{F}\neg\phi$, and $[[A]]\phi \stackrel{\triangle}{=} \neg\langle\langle A\rangle\rangle\neg\phi$. Given an ATL* formula ϕ and an atomic proposition q which is not used in ϕ, let $\phi[q/\varphi]$ denote the ATL* obtained by replacing every occurrence of the subformula φ in ϕ by q.

Let $cl(\phi)$ be the set of all subformulae of the ATL* formulae ϕ. Formally,

- $\phi \in cl(\phi)$;
- if $\neg\psi \in cl(\phi)$ or $\mathbf{X}\psi \in cl(\phi)$ or $\langle\langle A\rangle\rangle\psi \in cl(\phi)$, then $\psi \in cl(\phi)$
- if $\psi_1 \mathbf{U} \psi_2 \in cl(\phi)$, then $\psi_1, \psi_2, \mathbf{X}(\psi_1 \mathbf{U} \psi_2) \in cl(\phi)$.

The size $|\phi|$ of ϕ is defined as the size of the set $cl(\phi)$. In this paper, the semantics of ATL* is defined over PGSs. Let $\mathcal{P} = (P, \Gamma, \Delta, \lambda)$ be a PGS, ψ be an ATL* formula, π be a path of $\mathcal{P}$, the satisfiability relation $\mathcal{P}, \pi \models \psi$ is defined inductively as follows:

- $\mathcal{P}, \pi \models q$ iff $q \in \lambda(\pi_0)$;
- $\mathcal{P}, \pi \models \neg\phi$ iff $\mathcal{P}, \pi \not\models \phi$
- $\mathcal{P}, \pi \models \phi_1 \wedge \phi_2$ iff $\mathcal{P}, \pi \models \phi_1$ and $\mathcal{P}, \pi \models \phi_2$;
- $\mathcal{P}, \pi \models \mathbf{X}\phi$ iff $\mathcal{P}, \pi_{\geq 1} \models \phi$;
- $\mathcal{P}, \pi \models \phi_1 \mathbf{U} \phi_2$ iff there exists $i \geq 0$ such that $\mathcal{P}, \pi_{\geq i} \models \phi_2$ and for every $j : 0 \leq j < i$, $\mathcal{P}, \pi_{\geq j} \models \phi_1$;
- $\mathcal{P}, \pi \models \langle\langle A\rangle\rangle\phi$ iff there is an assignment $\upsilon_A \in V$ over the set A of agents such that for *every* $\pi' \in \prod_{\mathcal{P}}(\pi_0, \upsilon_A)$, $\mathcal{P}, \pi' \models \phi$.

Given a PGS $\mathcal{P}$, a configuration $c \in \mathcal{C}_{\mathcal{P}}$ and an ATL* formula ϕ, c satisfies ϕ, denoted by $\mathcal{P}, c \models \phi$, iff there exists a path $\pi \in \prod_{\mathcal{P}}(c)$ such that $\mathcal{P}, \pi \models \phi$.

ATL is a sub-logic of ATL* such that each occurrence of $\langle\langle A\rangle\rangle$ is followed immediately by an occurrence of **X**, or **U**, or **G**. More precisely, ATL formulae are defined by the following grammar,

$$\phi ::= q \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle A\rangle\rangle\mathbf{X}\phi \mid \langle\langle A\rangle\rangle\phi\mathbf{U}\phi \mid \langle\langle A\rangle\rangle\mathbf{G}\phi$$

where $q \in AP$ and $A \subseteq Ag$.

On the other hand, Linear Temporal Logic (LTL) is a sub-logic of ATL* such that the selective path quantifiers $\langle\langle A\rangle\rangle$ for $A \subseteq Ag$ are disallowed. Formally, LTL formulae are defined by the following grammar,

$$\phi ::= q \mid \neg\phi \mid \phi \wedge \phi \mid \mathbf{X}\phi \mid \phi\mathbf{U}\phi$$

where $q \in AP$.

Definition 3. *A* parity automaton $\mathcal{PA}$ *is a tuple* $(G, \Sigma, \theta, g^0, F)$ *where G is a finite set of states, Σ is the input alphabet, $\theta : G \times \Sigma \to 2^G$ is a transition function, $g^0 \in G$ is the initial state and $F : G \to \{0, ..., k\}$ is a rank function assigning each state $g \in G$ a priority $F(g)$, where k is some natural number called index.*

A *run* of $\mathcal{PA}$ over an ω-word $\alpha_0\alpha_1...$ from Σ^ω is a sequence of states $\pi = g_0 g_1 ...$ such that $g_0 = g^0$, and for every $i \geq 0$, $g_{i+1} \in \theta(g_i, \alpha_i)$. Let $inf(\pi)$ be the set of states visited infinitely often in π. A run π is *accepting* iff the smallest number of $\{F(g) \mid g \in inf(\pi)\}$ is *even*. $\mathcal{PA}$ is called *deterministic* if for every $(g, \alpha) \in G \times \Sigma$, $|\theta(g, \alpha)| \leq 1$. The transition function θ in a *deterministic parity automaton* (DPA) is written as $\theta : G \times \Sigma \to G$.

Theorem 1. *(Kupferman and Vardi 2001; Piterman 2007) For every LTL formula ϕ, we can construct a DPA with $2^{2^{\mathbf{O}(|\phi|)}}$ states and $2^{\mathbf{O}(|\phi|)}$ indices such that the DPA recognizes all of the ω-words satisfying ϕ.*

According to the definition of ATL*, it is easy to see the following proposition.

Proposition 1. *Given a PGS $\mathcal{P}$ and an ATL* formula ϕ, for every subformula $\langle\langle A\rangle\rangle\psi$ such that ψ is an LTL formula, if $C \subseteq \mathcal{C}_{\mathcal{P}}$ is the set of configurations that satisfy $\langle\langle A\rangle\rangle\psi$ and λ is extended as $q \in \lambda(c)$ for every $c \in C$ where q is a fresh atomic proposition, then for every configuration $c' \in \mathcal{C}_{\mathcal{P}}$, $\mathcal{P}, c' \models \phi$ iff $\mathcal{P}, c' \models \phi[q/\langle\langle A\rangle\rangle\psi]$.*

The syntax of AMC is given as follows.

Definition 4. *Given a set $\mathcal{Z}$ of propositional variables, AMC formulae are given by the following grammar:*

$$\begin{aligned}\phi ::= \quad & q \mid \neg q \mid Z \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle\langle A\rangle\rangle\boldsymbol{X}\phi \mid \\ & [[A]]\boldsymbol{X}\phi \mid \mu Z.\phi \mid \nu Z.\phi\end{aligned}$$

where $q \in AP$, $Z \in \mathcal{Z}$, $A \subseteq Ag$.

Fix a PGS $\mathcal{P} = (P, \Gamma, \Delta, \lambda)$, a *propositional valuation* is a function $\Omega : \mathcal{Z} \to 2^{\mathcal{C}_{\mathcal{P}}}$. The semantics of AMC is given by the denotation function $\|\circ\|^{\mathcal{P}}_{\Omega}$ that maps an AMC formula to a set of configurations of $\mathcal{P}$. Due to space restriction, its detailed definition is omitted here and we refer to, e.g., (Alur,

Henzinger, and Kupferman 2002). An AMC formula ϕ is *alternation-free* if for every subformula $\mu Z.\psi$ (resp. $\nu Z.\psi$) of ϕ, there is no subformula $\nu Z'.\varphi$ (resp. $\mu Z'.\varphi$) in ψ such that Z is a free variable of φ.

Definition 5 (Global Model Checking on PGSs)**.** *Given a PGS $\mathcal{P}$ and an ATL* or AMC formula ϕ, the* global *model checking problem is to compute the set of all the configurations $C \subseteq \mathcal{C}_\mathcal{P}$ such that for every $c' \in \mathcal{C}_\mathcal{P}$, $c' \in C$ iff $\mathcal{P}, c' \models \phi$.*

In this work, we consider the model checking problem with the labeling function $l : AP \to 2^{\mathcal{C}_\mathcal{P}}$ such that for every $q \in AP$, $l(q)$ is a *regular* set (technically, it is represented by an alternating multi-automaton; see below for definition). This is usually referred to as a *regular valuation* (Esparza, Kucera, and Schwoon 2003). l can be lifted to the function $\lambda_l : P \times \Gamma^* \to 2^{AP}$: for every $c \in \mathcal{C}_\mathcal{P}$, $\lambda_l(c) = \{q \in AP \mid c \in l(q)\}$.

Remark 1. *(Murano and Perelli 2015) already presented an example which is modeled as a PGS, as well as some properties that can be expressed in ATL*. We will not give examples here for the sake of space.*

Below we introduce some machinery which will be used in our model checking algorithms. In particular, model checking ATL* will be reduced to checking non-emptiness of alternating parity pushdown systems. Moreover, to *finitely* represent generally infinite sets of configurations of (alternating) pushdown systems, we use alternating multi-automata which are the "data structure" of the algorithm and play an essential role in algorithms for both ATL* and AMC.

Given a set X, let $\mathcal{B}^+(X)$ be the set of *positive Boolean formulae* over X. For a set $Y \subseteq X$ and a formula $\psi \in \mathcal{B}^+(X)$, Y satisfies ψ if assigning *true* to elements of Y and assigning *false* to elements of $X \setminus Y$ make ψ *true*.

Alternating Pushdown Systems

Definition 6. *An* Alternating Pushdown System (APDS) *is a tuple $\mathcal{P} = (P, \Gamma, \Delta)$, where P is a finite set of control states, Γ is a finite stack alphabet, and $\Delta : P \times \Gamma \to \mathcal{B}^+(P \times \Gamma^*)$ is a transition function that assigns to each element of $P \times \Gamma$ a positive Boolean formula over $P \times \Gamma^*$.*

For every set $\{\langle p_1, \omega_1\rangle, ..., \langle p_n, \omega_n\rangle\} \subseteq P \times \Gamma^*$ and every pair $\langle p, \gamma\rangle \in P \times \Gamma$, if $\{\langle p_1, \omega_1\rangle, ..., \langle p_n, \omega_n\rangle\}$ satisfies the positive Boolean formula $\Delta(\langle p, \gamma\rangle)$, we sometimes write $\langle p, \gamma\rangle \longrightarrow_\mathcal{P} \{\langle p_1, \omega_1\rangle, ..., \langle p_n, \omega_n\rangle\}$. If $\langle p, \gamma\rangle \longrightarrow_\mathcal{P} \{\langle p_1, \omega_1\rangle, ..., \langle p_n, \omega_n\rangle\}$, then $\langle p, \gamma\omega\rangle \Longrightarrow_\mathcal{P} \{\langle p_1, \omega_1\omega\rangle, ..., \langle p_n, \omega_n\omega\rangle\}$ for every $\omega \in \Gamma^*$. For every pair $\langle p, \gamma\rangle \in P \times \Gamma$, we suppose in this work that the Boolean formula $\Delta(\langle p, \gamma\rangle)$ is in the *disjunctive normal form*. The size $|\Delta|$ of Δ is defined as $\sum_{(p,\gamma)\in P\times\Gamma} |\Delta(p, \gamma)|$, where $|\Delta(p, \gamma)|$ denotes the number of satisfying sets of the Boolean formula $\Delta(p, \gamma)$.

A *run* ρ of the APDS $\mathcal{P}$ from a configuration $\langle p, \omega\rangle$ is a $\mathcal{C}_\mathcal{P}$-labeled tree (T_r, r) such that $r(\epsilon) = \langle p, \omega\rangle$, and for every node $t \in T_r$ with $r(t) = \langle p', \omega'\rangle$ and its children $t0, ..., tn$, it must be the case that $\langle p', \omega'\rangle \Longrightarrow_\mathcal{P} \{\langle p'_0, \omega'_0\rangle, ..., \langle p'_n, \omega'_n\rangle\}$ where $r(ti) = \langle p'_i, \omega'_i\rangle$ for every $i : 0 \leq i \leq n$. W.l.o.g., we assume that all of the runs of APDSs are infinite. Given a path π of the run ρ, let $inf(\pi)$ denote the set of control states appearing infinitely often in $r(\pi)$.

Alternating Multi-Automata

Definition 7. *(Bouajjani, Esparza, and Maler 1997) Let $\mathcal{P} = (P, \Gamma, \Delta, F)$ be an APDS. An Alternating Multi-Automaton* (AMA) *is a tuple $\mathcal{M} = (S, \Gamma, \delta, I, S_f)$, where S is a finite set of states with $S \supseteq P$, Γ is the input alphabet, $\delta : (S \times \Gamma) \to \mathcal{B}^+(S)$ is a transition function, $I \subseteq P$ is a finite set of initial states, $S_f \subseteq S$ is a finite set of final states.*

As before, for a set of states $\{s_1, ..., s_n\} \subseteq S$, if $\{s_1, ..., s_n\}$ satisfies $\delta(s, \gamma)$, we will sometimes write $s \xrightarrow{\gamma} \{s_1, ..., s_n\}$ instead. We define the relation $\longrightarrow_\delta \subseteq S \times \Gamma^* \times 2^S$ as the least relation such that the following conditions hold:

- $s \xrightarrow{\epsilon}_\delta \{s\}$ for every $s \in S$;
- $s \xrightarrow{\gamma\omega}_\delta \bigcup_{i=1}^n S_i$ if $s \xrightarrow{\gamma} \{s_1, ..., s_n\}$ and $s_i \xrightarrow{\omega}_\delta S_i$ for every $i : 1 \leq i \leq n$.

The AMA $\mathcal{M}$ *accepts* a configuration $\langle p, \omega\rangle$ if there exists $S' \subseteq S_f$ such that $p \xrightarrow{\omega}_\delta S'$ and $p \in I$. Let $\mathcal{L}(\mathcal{M})$ denote the set of all the configurations accepted by $\mathcal{M}$.

Proposition 2. *(Cachat 2002) Let $\mathcal{M} = (S, \Gamma, \delta, I, S_f)$ be an AMA. Deciding whether a configuration $\langle p, \omega\rangle$ with $p \in S$ and $\omega \in \Gamma^*$ is accepted by $\mathcal{M}$ or not can be done in $\mathbf{O}(|S| \cdot |\delta| \cdot |\omega|)$ time and $\mathbf{O}(|S|)$ space.*

3 ATL* Model Checking on PGSs

In this section, we propose an automata-theoretic approach to the global model checking of ATL* on PGSs with regular valuations. For this purpose, we need alternating pushdown systems with parity acceptance conditions.

An *Alternating Parity Pushdown System (APPDS)* $\mathcal{PP} = (P, \Gamma, \Delta, F)$ is an APDS (P, Γ, Δ) with the parity acceptance condition given by $F : P \to \{0, ..., k\}$. A path π in a run ρ of the APPDS $\mathcal{PP}$ is *accepting* if and only if the smallest number in $\{F(p) \mid p \in inf(\pi)\}$ is *even*. A run ρ in the APPDS $\mathcal{PP}$ is *accepting* if and only if all paths of ρ are accepting. Let $\mathcal{L}(\mathcal{PP})$ denote the set of all configurations from which the APPDS $\mathcal{PP}$ has an accepting run.

Theorem 2. *(Hague and Ong 2009) For an APPDS $\mathcal{PP} = (P, \Gamma, \Delta, F)$, an AMA $\mathcal{M}$ with $\mathbf{O}(|P|)$ states and $\mathbf{O}(|P| \cdot |\Gamma| \cdot 2^{|P|})$ transition rules can be computed in $2^{\mathbf{O}(k|P|)}$ time and space such that $\mathcal{L}(\mathcal{M}) = \mathcal{L}(\mathcal{PP})$, where k is the index of $\mathcal{PP}$.*

Note that the nonemptiness problem for the (general) alternating parity pushdown tree automata (APPTA), of which APPDS can be considered as a special case, is undecidable. However, the APPDS considered here has always the same stack content over the same direction of the trees, which is essential for the decidability results. See also (Aminof et al. 2013) for further discussions on this point.

Throughout this section, let $\mathcal{P} = (P, \Gamma, \Delta, \lambda_l)$ be a PGS and ϕ be an ATL* formula. Without loss of generality, we assume that ϕ is a Boolean combination of formulae of the

form $\langle\langle A\rangle\rangle\psi$, where ψ is an ATL* formula. Therefore, an AMA $\mathcal{M}^{\phi}$ can be computed via Boolean operations on AMAs. In the following, we will demonstrate how to compute, for each subformula $\langle\langle A\rangle\rangle\psi$ of ϕ, an AMA $\mathcal{M}^{\langle\langle A\rangle\rangle\psi}$ in a bottom-up approach to recognize the set of configurations of $\mathcal{P}$ satisfying $\langle\langle A\rangle\rangle\psi$.

Assume that for each proper subformula $\langle\langle A'\rangle\rangle\psi'$ of ψ, an AMA $\mathcal{M}^{\langle\langle A'\rangle\rangle\psi'} =$

$$(S^{\langle\langle A'\rangle\rangle\psi'}, \Gamma^{\langle\langle A'\rangle\rangle\psi'}, \delta^{\langle\langle A'\rangle\rangle\psi'}, I^{\langle\langle A'\rangle\rangle\psi'}, S_f^{\langle\langle A'\rangle\rangle\psi'})$$

has been computed so that $\mathcal{L}(\mathcal{M}^{\langle\langle A'\rangle\rangle\psi'}) = \{c \in \mathcal{C}_{\mathcal{P}} \mid \mathcal{P}, c \models \langle\langle A'\rangle\rangle\psi'\}$. Moreover, since AMAs are closed under complementation, we assume that $\mathcal{M}^{\neg\langle\langle A'\rangle\rangle\psi'}$ has also been computed to recognize $\mathcal{C}_{\mathcal{P}} \setminus \mathcal{L}(\mathcal{M}^{\langle\langle A'\rangle\rangle\psi'})$. Then let $\langle\langle A\rangle\rangle\psi_1$ be the ATL* formula obtained from $\langle\langle A\rangle\rangle\psi$ as follows: For each proper subformulae $\langle\langle A'\rangle\rangle\psi'$ of ψ, replace each occurrence of $\langle\langle A'\rangle\rangle\psi'$ with a fresh atomic proposition $q_{\langle\langle A'\rangle\rangle\psi'}$. Let AP' denote the union of AP and these fresh atomic propositions. Clearly ψ_1 is an LTL formula over AP'. Therefore, by Proposition 1 the global model checking of $\langle\langle A\rangle\rangle\psi$ over $\mathcal{P}$ with the regular valuation l can be reduced to the global model checking of $\langle\langle A\rangle\rangle\psi_1$ over $\mathcal{P}' = (P, \Gamma, \Delta, \lambda_{l'})$, where l' is the regular valuation extended from l by assigning $\mathcal{L}(\mathcal{M}^{\langle\langle A'\rangle\rangle\psi'})$ to the fresh atomic propositions $q_{\langle\langle A'\rangle\rangle\psi'}$.

In the following, we will construct an APPDS $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}$ from $\mathcal{P}'$ and $\langle\langle A\rangle\rangle\psi_1$ so that $\mathcal{L}(\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}) = \{c \in \mathcal{C}_{\mathcal{P}'} \mid \mathcal{P}', c \models \langle\langle A\rangle\rangle\psi_1\}$. From Theorem 2, it follows that the desired AMA $\mathcal{M}^{\langle\langle A\rangle\rangle\psi}$ can be constructed from $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}$.

It remains to construct $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}$. To do this, we first construct a DPA $\mathcal{PA}_{\psi_1} = (G, \Sigma, \theta, g^0, F)$ that recognizes all the ω-words satisfying ψ_1 (cf. Theorem 1), where $\Sigma = 2^{AP'}$. As the next step, intuitively the *existential* selection of strategies of the agents from A for model checking $\langle\langle A\rangle\rangle\psi_1$ on $\mathcal{P}'$ is represented by the *disjunctions* in the transition rules of $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}$. On the other hand, once the strategies of the agents from A are fixed, all the paths resulting from the selection of strategies of the agents outside A, should satisfy ψ_1. This *universal* constraint is specified by the *conjunctions* in the transition rules of $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}$. The runs of $\mathcal{PA}_{\psi_1}$ on these paths are mimicked via the runs of the APPDS $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1}$, where the satisfaction (resp. dissatisfaction) of an atomic proposition $q' \in AP'$ on a configuration is verified by a new thread which keeps popping the stack and simulates the run of the AMA $\mathcal{M}^{q'}$ (resp. $\mathcal{M}^{\neg q'}$) over the configuration.

Suppose the DPA $\mathcal{PA}_{\psi_1} = (G, \Sigma, \theta, g^0, F)$ constructed from ψ_1 has index k. Moreover, for every $q' \in AP'$, we assume that the AMA $\mathcal{M}^{q'} = (S^{q'}, \Gamma, \delta^{q'}, I^{q'}, S_f^{q'})$ and its complement $\mathcal{M}^{\neg q'} = (S^{\neg q'}, \Gamma, \delta^{\neg q'}, I^{\neg q'}, S_f^{\neg q'})$ have been computed. Although the states from P may occur in different AMAs, for our purpose, we assume that each occurrence of $p \in P$ in different $\mathcal{M}^{q'}$ or $\mathcal{M}^{\neg q'}$ carries a unique name, for instance, it is decorated by q' (resp. $\neg q'$), denoted by $p^{q'}$ (resp. $p^{\neg q'}$).

We construct $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1} = (P', \Gamma, \Delta', F')$ as follows.

- $P' = (P \times G) \cup \bigcup_{q' \in AP'} S^{q'} \cup S^{\neg q'}$;
- $F' : P' \to \{0, ..., k\}$ such that for every $p' \in P'$,

$$F'(p') = \begin{cases} F(g), & \text{if } p' = [p, g] \in P \times G, \\ 0, & \text{otherwise;} \end{cases}$$

- Δ' is the smallest transition function satisfying the following conditions:
  - i. for each pair $(p, \gamma) \in P \times \Gamma$, $g \in G, \alpha \subseteq AP'$, $\Delta'(\langle [p, g], \gamma\rangle) = \bigwedge_{q' \in \alpha} \langle p^{q'}, \gamma\rangle \wedge \bigwedge_{q' \in AP' \setminus \alpha} \langle p^{\neg q'}, \gamma\rangle \wedge \bigvee_{f: A \to Ac} \big(\bigwedge_{\langle p', \omega\rangle \in succ_f(p, \gamma)} \langle [p', \theta(g, \alpha)], \omega\rangle \big)$;
  - ii. for every $s \xrightarrow{\gamma} \{s_1, ..., s_n\} \in \bigcup_{q' \in AP'} \delta^{q'} \cup \delta^{\neg q'}$, $\Delta'(\langle s, \gamma\rangle) = \bigwedge_{1 \le i \le n} \langle s_i, \epsilon\rangle$;
  - iii. for every $s \in \bigcup_{q' \in AP'} S_f^{q'} \cup S_f^{\neg q'}$, $\Delta'(\langle s, \bot\rangle) = \langle s, \bot\rangle$.

Theorem 3. *Let $\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1} = (P', \Gamma, \Delta', F')$ be constructed as above. Then for every configuration $\langle p, \omega\rangle \in \mathcal{C}_{\mathcal{P}'}$, $\mathcal{P}', \langle p, \omega\rangle \models \langle\langle A\rangle\rangle\psi_1$ iff $\langle [p, g^0], \omega\rangle \in \mathcal{L}(\mathcal{PP}^{\langle\langle A\rangle\rangle\psi_1})$. Moreover, $|P'|$ and $|\Delta'|$ are doubly exponential of $|\psi_1|$, and polynomial of the size of $\mathcal{P}'$ and the size of $\mathcal{M}^{q'}$ for $q' \in AP'$. In addition, the index k is exponential of $|\psi_1|$.*

We deduce from Theorem 3 and Theorem 2 that the size of $\mathcal{M}^{\phi}$ is triply exponential in $|\phi|$ and polynomial in the size of $\mathcal{P}$. From Proposition 2, we obtain the main result of this section.

Theorem 4. *The model checking problem for ATL* over PGSs is 3EXPTIME-complete.*

The lower bound follows from a reduction from two-player pushdown games with winning conditions specified by LTL formulae, which was shown to be 3EXPTIME-complete in (Löding, Madhusudan, and Serre 2004) [1].

An instance $\mathcal{G}$ of two-player pushdown games with winning conditions specified by LTL formulae is a tuple $(\mathcal{P}, \lambda_l, P_0, P_1, \phi)$, where

- $\mathcal{P} = (P, \Gamma, \Delta)$ is a pushdown system,
- $\lambda : P \times \Gamma \to 2^{AP}$ is a function that assigns each pair $\langle p, \gamma\rangle$ a subset of AP,
- (P_0, P_1) forms a partition of P, called respectively the set of states for player 0 and player 1,
- and ϕ is an LTL formula over AP, called the winning condition for player 0.

Plays of $\mathcal{G}$ are defined as usual. A play of $\mathcal{G}$, say $(p_0, \gamma_0\omega_0)(p_1, \gamma_1\omega_1)\ldots$, is *winning for player 0* if the ω-word $\lambda(\langle p_0, \gamma_0\rangle)\lambda(\langle p_1, \gamma_1\rangle)\ldots$ satisfies ϕ. Winning strategies and winning regions are then defined as usual.

[1] We could also prove the lower bound by a more involved reduction from the membership problem of 2EXPSPACE alternating Turing machines.

In the following, we construct in polynomial time a PGS $\mathcal{P}' = (P \cup \{p_\perp\}, \Gamma, \Delta', \lambda')$ and an ATL* formula ϕ' such that for each configuration c of $\mathcal{P}$, $\mathcal{P}, c \models \phi$ iff $\mathcal{P}', c \models \phi'$.

- $Ag = \{ag_0, ag_1\}$ where ag_0, ag_1 correspond to player 0 and 1 of $\mathcal{P}$ respectively.
- $Ac = \{1, \ldots, K\}$, where K is the maximum number k such that $\Delta(\langle p, \gamma\rangle) = \{\langle p_1, \omega_1\rangle, \ldots, \langle p_k, \omega_k\rangle\}$ for some $p \in P$ and $\gamma \in \Gamma$.
- For each $\langle p, \gamma\rangle \in P_0 \times \Gamma$ such that $\Delta(\langle p, \gamma\rangle) = \{\langle p_1, \omega_1\rangle, \ldots, \langle p_k, \omega_k\rangle\}$, and each $d \in Dc$, if $d(ag_0) = i \in \{1, \ldots, k\}$, then $\Delta'(\langle p, \gamma\rangle, d) = \langle p_i, \omega_i\rangle$, otherwise, $\Delta'(\langle p, \gamma\rangle, d) = \langle p_\perp, \gamma\rangle$. Note that here the transitions of Δ' are determined by the actions of the agent ag_0, no matter what actions taken by the agent ag_1.
- For each $\langle p, \gamma\rangle \in P_1 \times \Gamma$ such that $\Delta(\langle p, \gamma\rangle) = \{\langle p_1, \omega_1\rangle, \ldots, \langle p_k, \omega_k\rangle\}$, and each $d \in Dc$, if $d(ag_1) = i \in \{1, \ldots, k\}$, then $\Delta'(\langle p, \gamma\rangle, d) = \langle p_i, \omega_i\rangle$, otherwise, $\Delta'(\langle p, \gamma\rangle, d) = \langle p_\perp, \gamma\rangle$. Note that here the transitions of Δ' are determined by the actions of the agent ag_1, no matter what actions taken by the agent ag_0.
- For each configuration $\langle p, \gamma\omega\rangle$ of $\mathcal{P}'$ such that $p \in P$, $\lambda'(\langle p, \gamma\omega\rangle) = \lambda(\langle p, \gamma\rangle)$. For each configuration $\langle p_\perp, \gamma\omega\rangle$ of $\mathcal{P}'$, $\lambda'(\langle p_\perp, \gamma\omega\rangle) = q_\perp$.
- The LTL formula $\phi' = \langle\langle\{ag_0\}\rangle\rangle(G\neg q_\perp \wedge \phi)$.

4 Alternating-time μ-Calculus

In this section, we propose a global model checking algorithm for the alternating-time μ-calculus on PGSs.

Given a PGS $\mathcal{P} = (P, \Gamma, \Delta, \lambda_{l'})$, an AMC formula ϕ and a propositional valuation Ω for the free variables in ϕ. The algorithm constructs an AMA representing the denotation of ϕ with respect to $\mathcal{P}$, i.e., $\|\phi\|^{\mathcal{P}}_{\Omega}$.

The algorithm follows closely the saturation method for the modal μ-calculus over pushdown systems (Hague and Ong 2011). In particular, for the least and greatest fixpoints, we shall apply the *projection* function to ensure termination. Because of the similarity of syntax, most procedures, as well as the correctness proof, are (almost) identical to those in (Hague and Ong 2011). However, to handle the path selective quantifier, we need to adapt those procedures for the box and diamond modalities of the modal μ-calculus, i.e., for $\psi = \langle\langle A\rangle\rangle\mathbf{X}\phi$ and $\psi' = [[A]]\mathbf{X}\phi'$.

Suppose we have generated an AMA $(S_1, \Gamma, \delta_1, I_1, S^1_f)$ for ϕ (resp. ϕ'), our task now is to generate a new AMA for ψ (resp. ψ'). This can be defined as

$$(S_1 \cup I, \Gamma, \delta_1 \cup \delta', I, S^1_f),$$

where $I = \{[p, \psi] \mid p \in P\}$ (resp. $I = \{[p, \psi'] \mid p \in P\}$), and

$$\delta'([p, \psi], \gamma) = \bigvee_{f: A \to Ac} \bigwedge_{\langle p', \omega\rangle \in succ_f(p,\gamma)} \bigwedge_{s \in Q_{p'\omega}} s$$

(resp.

$$\delta'([p, \psi'], \gamma) = \bigwedge_{f: A \to Ac} \bigvee_{\langle p', \omega\rangle \in succ_f(p,\gamma)} \bigwedge_{s \in Q_{p'\omega}} s\ )$$

where $p' \xrightarrow{\omega}_{\delta_1} Q_{p'\omega}$.

The constructions for the other operators of the AMC follow those in (Hague and Ong 2011) accordingly.

Theorem 5. *The model checking problem for AMC on PGSs is EXPTIME-complete.*

The hardness follows from the hardness of model checking problems for the alternation-free modal μ-calculus over pushdown systems, which is EXPTIME-complete (Walukiewicz 2001), and the obvious observations that alternation-free μ-calculus is a fragment of (alternation-free) AMC, and pushdown systems are a special class of PGSs (i.e., when $|Ag| = |Ac| = 1$).

Indeed we even have:

Corollary 1. *The model checking problem for the* alternation-free *AMC on PGSs is EXPTIME-complete.*

Since each ATL formula can be translated into an equivalent alternation-free AMC formula in linear time, we obtain the following result.

Corollary 2. *The model checking problem for ATL on PGSs is EXPTIME-complete.*

The lower bound follows from the fact that the control state reachability problem of APDSs is EXPTIME-complete (Suwimonteerabuth, Schwoon, and Esparza 2006)[2]: We can reduce this problem to the model checking problem for a simple ATL formula $\langle\langle A\rangle\rangle\mathbf{F}q$ on PGSs.

5 Related Work

Model checking for LTL/CTL on pushdown systems were well studied in the literature and were used to verify infinite-state closed systems such as sequential programs with recursion, see (Carayol and Hague 2014) for a survey. To verify infinite-state open systems, solving two-player games or module checking on pushdown systems were studied in several works (Walukiewicz 2001; Hague and Ong 2009; Löding, Madhusudan, and Serre 2004; Serre 2003; Aminof et al. 2013; Bozzelli, Murano, and Peron 2010). However, they are incomparable to multi-agent pushdown systems, as discussed in (Murano and Perelli 2015). Model checking techniques were extended to verify *finite-state* multi-agent systems (Bourahla and Benmohamed 2005; Bulling and Jamroga 2011; Jamroga and Murano 2015; Cermák, Lomuscio, and Murano 2015). Moreover, various extensions of alternating time temporal logics have been considered, for instance, ATL with strategy contexts (Lopes, Laroussinie, and Markey 2010), ATL with strategy interactions (Wang, Schewe, and Huang 2015), strategy logic (Chatterjee, Henzinger, and Piterman 2010), ATL with probabilistic extensions (Chen et al. 2013). Again, these works are restricted to finite-state multi-agent systems.

The most closely related work is (Murano and Perelli 2015) which proposes an automata-theoretic approach to model checking for ATL* on PGSs which are *infinite-state*

[2]Although the result in (Suwimonteerabuth, Schwoon, and Esparza 2006) is for the backward reachability problem, it is not hard to see that the proof therein can be slightly adapted to show that the control state reachability problem is EXPTIME-complete.

multi-agent systems. The main differences between (Murano and Perelli 2015) and our work have been discussed extensively in the introduction.

6 Conclusion

In this paper, we proposed a novel top-down approach for the global model checking of ATL* and AMC on PGSs with regular valuations. We show that they are 3EXPTIME-complete and EXPTIME-complete respectively; the latter complexity bound holds for the alternation-free fragment of AMC and ATL. These algorithms are saturation-based which we believe are crucial for efficient implementation.

Future work includes, apart from providing tool support and case studies, investigations of other logics (as listed partially in the related work) over PGSs, in particular their efficient model checking algorithms and complexity.

7 Acknowledgments

The authors are grateful to the anonymous referees for their constructive comments. Taolue Chen is partially supported by the ARC Discovery Project (DP160101652), the Singapore Ministry of Education AcRF Tier 2 grant (MOE2015-T2-1-137), and an oversea grant from the State Key Laboratory of Novel Software Technology, Nanjing Unviersity. Fu Song is partially supported by Shanghai Pujiang Program (No. 14PJ1403200), Shanghai ChenGuang Program (No. 13CG21), and NSFC Projects (No. 61402179, 61532019, and 91418203). Zhilin Wu is partially supported by the NSFC projects (No. 61272135, 61472474, and 61572478).

References

Alur, R.; Henzinger, T. A.; and Kupferman, O. 2002. Alternating-time temporal logic. *J. ACM* 49(5):672–713.

Aminof, B.; Legay, A.; Murano, A.; Serre, O.; and Vardi, M. Y. 2013. Pushdown module checking with imperfect information. *Inf. Comput.* 223:1–17.

Baier, C., and Katoen, J. P. 2008. *Principles of model checking*. MIT Press.

Bouajjani, A.; Esparza, J.; and Maler, O. 1997. Reachability Analysis of Pushdown Automata: Application to Model Checking. In *CONCUR'97*. LNCS 1243.

Bourahla, M., and Benmohamed, M. 2005. Model checking multi-agent systems. *Informatica (Slovenia)* 29(2):189–198.

Bozzelli, L.; Murano, A.; and Peron, A. 2010. Pushdown module checking. *Formal Methods in System Design* 36(1):65–95.

Bulling, N., and Jamroga, W. 2011. Alternating epistemic mu-calculus. In *IJCAI'11*, 109–114.

Cachat, T. 2002. Symbolic strategy synthesis for games on pushdown graphs. In *ICALP'02*, 704–715.

Carayol, A., and Hague, M. 2014. Saturation algorithms for model-checking pushdown systems. In *AFL'14*, 1–24.

Cermák, P.; Lomuscio, A.; and Murano, A. 2015. Verifying and synthesising multi-agent systems against one-goal strategy logic specifications. In *AAAI'15*, 2038–2044.

Chatterjee, K.; Henzinger, T. A.; and Piterman, N. 2010. Strategy logic. *Inf. Comput.* 208(6):677–693.

Chen, T.; Forejt, V.; Kwiatkowska, M. Z.; Parker, D.; and Simaitis, A. 2013. Automatic verification of competitive stochastic systems. *Formal Methods in System Design* 43(1):61–92.

Clarke, E. M., and Emerson, E. A. 1981. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Proceeding of Workshop on Logics of Programs, Yorktown Heights, New York*, 52–71.

de Alfaro, L.; Henzinger, T. A.; and Majumdar, R. 2001. From verification to control: Dynamic programs for omega-regular objectives. In *LICS'01*, 279–290.

Esparza, J.; Kucera, A.; and Schwoon, S. 2003. Model checking LTL with regular valuations for pushdown systems. *Inf. Comput.* 186(2):355–376.

Hague, M., and Ong, C. L. 2009. Winning regions of pushdown parity games: A saturation method. In *CONCUR'09*, 384–398.

Hague, M., and Ong, C. L. 2011. A saturation method for the modal μ-calculus over pushdown systems. *Inf. Comput.* 209(5):799–821.

Jamroga, W., and Murano, A. 2015. Module checking of strategic ability. In *AAMAS'15*, 227–235.

Kozen, D. 1983. Results on the propositional mu-calculus. *Theor. Comput. Sci.* 27:333–354.

Kupferman, O., and Vardi, M. Y. 2001. Weak alternating automata are not that weak. *ACM Trans. Comput. Log.* 2(3):408–429.

Löding, C.; Madhusudan, P.; and Serre, O. 2004. Visibly pushdown games. In *FSTTCS'04*, 408–420.

Lopes, A. D. C.; Laroussinie, F.; and Markey, N. 2010. ATL with strategy contexts: Expressiveness and model checking. In *FSTTCS'10*, 120–132.

Murano, A., and Perelli, G. 2015. Pushdown multi-agent system verification. In *IJCAI'15*, 1090–1097.

Muscettola, N.; Nayak, P. P.; Pell, B.; and Williams, B. C. 1998. Remote agent: To boldly go where no AI system has gone before. *Artif. Intell.* 103(1-2):5–47.

Piterman, N., and Vardi, M. Y. 2004. Global model-checking of infinite-state systems. In *CAV'04*, 387–400.

Piterman, N. 2007. From nondeterministic Büchi and streett automata to deterministic parity automata. *Logical Methods in Computer Science* 3(3).

Pnueli, A. 1977. The temporal logic of programs. In *FOCS'77*, 46–57.

Serre, O. 2003. Note on winning positions on pushdown games with [omega]-regular conditions. *Inf. Process. Lett.* 85(6):285–291.

Suwimonteerabuth, D.; Schwoon, S.; and Esparza, J. 2006. Efficient algorithms for alternating pushdown systems with an application to the computation of certificate chains. In *ATVA'06*, 141–153.

Walukiewicz, I. 2001. Pushdown processes: Games and model-checking. *Inf. Comput.* 164(2):234–263.

Wang, F.; Schewe, S.; and Huang, C. 2015. An extension of ATL with strategy interaction. *ACM Trans. Program. Lang. Syst.* 37(3):9.