Parity and Exploration Games on Infinite Graphs*

Hugo Gimbert

Université Paris 7, LIAFA, case 7014
2, place Jussieu
75251 Paris Cedex 05, France
Hugo.Gimbert@liafa.jussieu.fr

Abstract. This paper examines two players' turn-based perfect-information games played on infinite graphs. Our attention is focused on the classes of games where winning conditions are boolean combinations of the following two conditions: (1) the first one states that an infinite play is won by player 0 if during the play infinitely many different vertices were visited, (2) the second one is the well known parity condition generalized to a countable number of priorities.
We show that, in most cases, both players have positional winning strategies and we characterize their respective winning sets. In the special case of pushdown graphs, we use these results to show that the sets of winning positions are regular and we show how to compute them as well as positional winning strategies in exponential time.

1 Introduction

Two-player games played on graphs have attracted a lot of attention in computer science. In verification of reactive systems it is natural to see the interactions between a system and its environment as a two-person game [19,9], in control theory the problem of controller synthesis amounts often to finding a winning strategy in an associated game [1].

Depending on the nature of the examined systems various types of two-player games are considered. The interactions between players can be turn-based [23,19] or concurrent [7,8], finite like in reachability games or infinite like in parity or Muller games, the players can have perfect or imperfect information about the play. Moreover, the transitions may be deterministic or stochastic [6,8] and finally the system itself can be finite or infinite.

Another source of diversity comes from players' objectives, i.e. winning conditions.

Our work has as a framework turn-based perfect information infinite games on pushdown graphs. The vertices of such graphs correspond to configurations

* This research was supported by European Research Training Network: Games and Automata for Synthesis and Validation.

of a pushdown automaton and edges are induced by push-down automaton transitions. The interest in such games comes, at least in part, from practical considerations, pushdown systems can model, to some extent, recursive procedure calls. On the other hand, pushdown graphs constitute one of the simplest class of infinite graphs that admit non trivial positive decidability results and since the seminal paper of Muller and Schupp [14] many other problems are shown to be decidable for this class [2,13,5,18,3,22,4,17].

Let us describe briefly a play of such a game. The set of vertices is partitioned into two sets: vertices belonging to player 0 and vertices belonging to his adversary 1. Initially, a pebble is placed on a vertex. At each turn the owner of the vertex with the pebble chooses a successor vertex and moves the pebble onto it. Then the owner of this new vertex proceeds in the same way, and so on. The successive pebble positions form an infinite path in the graph, this is the resulting play.

In this framework, different objectives have been studied. Such an objective is described in general as the set of infinite plays that are winning for player 0, and it is called a winning condition. A lot of attention has been given to the case where this set is regular, which gives rise to Müller and parity winning conditions [23,22,19] which lie on the level Δ_2 of the Borel hierarchy. However, recently Cachat et al. [5], presented a new winning condition of Borel complexity Σ_3 which still remains decidable. This Σ_3-condition specifies that player 0 wins a play if there is no vertex visited infinitely often. Yet another condition, *unboundedness*, was introduced by Bouquet et al. [3]. The unboundedness condition states that player 0 wins a play if the corresponding sequence of stack heights is unbounded. Obviously the conditions of [5] and [3] are tightly related, if no configuration of the push-down system is visited infinitely often then the stack is unbounded. The converse can be established as well if the winning strategies are memoryless, i.e. do not depend on the past.

In this paper, we first transfer the condition of [3] to arbitrary infinite graphs of finite degree. In the context of arbitrary infinite graphs we examine *Exploration* condition which states that a play is won by player 0 if the pebble visits an infinite number of different vertices. Obviously for the particular case of pushdown graphs this gives the same condition as [3]. In fact we go a step further and consider the games whose winning conditions are boolean combinations of Exploration condition and of the classical parity condition. We note respectively $\mathit{Exp} \cup \mathit{Parity}$ and $\mathit{Exp} \cap \mathit{Parity}$ the games obtained by taking the union and the intersection of Exploration and Parity conditions.

We also consider a particular extension of the classical Parity condition to the case with an infinite number of priorities and denote it Parity_∞ (see also [11] for another approach to parity games with an infinity of priorities).

We prove the following results in the context of the games over any infinite graphs:

- Both players have positional winning strategies for the game with the winning condition $\mathit{Exp} \cup \mathit{Parity}$, including the case where there is an infinite number of priorities.

– In the case where there are finitely many priorities, player 1 has also a winning positional strategy in the game where the winning condition for player 0 is of type $Exp \cap Parity$. Moreover, we can easily characterize the set of winning positions of player 0.

Even if general results concerning winning strategies over arbitrary infinite graphs are of some interest we are much more interested in decidability results for the special case of pushdown graphs. In the case where the game graph is a pushdown graph, we prove for both types of games $Exp \cup Parity$ and $Exp \cap Parity$ that the sets of winning configurations (positions) for player 0 (and also for player 1) are regular subsets of $Q\Gamma^*$ where Q is the set of states of pushdown system and Γ is the stack alphabet. We provide also an algorithm for computing a Büchi automaton with $2^{\mathcal{O}(d^2|Q|^2+|\Gamma|)}$ states recognizing those winning sets, where d is the number of priorities of the underlying parity game and Q and Γ are as stated above. Moreover, we show that for both games and both players, the set of winning positional strategies is regular and recognized by an alternating Büchi automaton. In the case of the $Exp \cup Parity_d$ game, this automaton has $\mathcal{O}(d|Q|^2 + |\Gamma|)$ states whereas in the case of the $Exp \cap Parity_d$ game, it has $\mathcal{O}(d^2|Q|^2 + d|\Gamma|)$ states

These results constitute an extension of the results of [5,3,22,18,20]: The papers [22,20,18] examine only *Parity* conditions with a finite number of priorities for pushdown games. Bouquet et al. [3] were able to extend the decidability results to the games with the winning condition of the form $Exp \cup Buchi$ or $Exp \cap Buchi$, i.e. union and intersections of Büchi condition with Exploration condition. However this class of conditions is not closed under boolean operations (intersecting Büchi and co-Büchi conditions with an Exploration condition is not in this class). In our paper we go even further since we allow boolean combinations of *Exp* conditions with parity conditions. Since parity conditions, after appropriate transformations, are closed under boolean operations we show in fact that it is decidable to determine a winner for the smallest class of conditions containing Exploration and Büchi conditions and closed under finite boolean operations.

For computing the winning sets and the winning strategies, we make use of tree automata techniques close to the one originated in the paper of Vardi [20] and applied in [16,12]. This is a radical departure from the techniques applied in [21,22,3,18] which are based on game-reductions.

This paper is organized as follows. In the first part, we introduce some basic definitions and the notions of Exploration and Parity games. In the second part, we prove the results concerning the winning strategies for the games $Exp \cup Parity_\infty$ and $Exp \cap Parity$, and make some comments about the $Parity_\infty$ game. In the third part, we describe the construction of automata computing the winning sets and the winning positional strategies. Due to space limitation, most proofs are omitted and can be found in the full version [10].

2 Parity and Exploration Games

In this section, we present basic notions about games and we define different winning conditions.

2.1 Generalities

The games we study are played on oriented graphs of finite degree, with no dead-ends, whose vertex set is partitioned between the vertices of player 0 and the vertices of player 1. Such a graph is called an arena. At the beginning of a play, a pebble is put on a vertex. During the play, the owner of the vertex with the pebble moves it to one of the successors vertices. A play is the infinite path visited by the pebble. A winning condition determines which player is the winner. Here follows the formal description of these notions.

Notations. Let $G = (V, E)$ be an oriented graph with the set $E \subset V \times V$ of edges. Given a vertex v, vE denotes the set of successors of v, $vE = \{w \in V : (v, w) \in E\}$, whereas Ev is the set of predecessors of v. For a set $H \subseteq E$ of edges, $Dom(H)$, the domain of H, denotes the set of the vertices adjacent to edges of H.

Parity arenas. An arena is a tuple (V, V_0, V_1, E), where (V, E) is a graph of finite degree with no dead-ends and (V_0, V_1) is a partition of V. Let $i \in \{0, 1\}$ be a player. V_i is the set of vertices of player i. We will often say that $G = (V, E)$ itself is an arena, when the partition (V_0, V_1) is obvious. An infinite path in G is called a play, whereas a finite path in G is called a finite play. When the vertices of G are labeled with natural numbers with a map $\phi : V \to \mathbb{N}$, G is said to be a parity arena.

Winning Conditions and Games. A winning condition determines the winner of a play. Formally, it is a subset $\mathrm{Vic} \subseteq V^\omega$ of the set of infinite plays. A game is a couple (G, Vic) made of an arena and a winning condition. Often, when the arena G is obvious, we will say that Vic itself is a game. A play $p \in V^\omega$ is won by player 0 if $p \in \mathrm{Vic}$. Otherwise, if $p \notin \mathrm{Vic}$, it is said to be won by player 1. Vic is said to be *concatenation-closed* if $V^* \,\mathrm{Vic} = \mathrm{Vic}$.

Strategies, Winning Strategies and Winning Sets. Depending on the finite path followed by the pebble, a strategy allows a player to choose between a restricted number of successor vertices. Let $i \in \{0, 1\}$ be a player. Formally, a strategy for player i is a map σ, which associates to any finite play $v_0 \cdots v_n$ such that $v_n \in V_i$ a nonempty subset $\sigma(v_0 ... v_n) \subseteq v_n E$. A play $p = (v_n)_{n \in \mathbb{N}} \in V^\omega$ is said to be consistent with σ if, for any n such that $v_n \in V_i$, $v_{n+1} \in \sigma(v_0 \cdots v_n)$. Given a subset $X \subseteq V$ of the vertices, A strategy for player i is said to be winning the game (G, Vic) on X if any infinite play starting in X and consistent with this strategy is won by player i. If there exists such a strategy, we say that player i wins (G, Vic) on X. If $X = V$, we simply say that i wins (G, Vic). The winning set of player i is the greatest set of vertices such that i wins Vic on this set.

Positional Strategies. With certain strategies, the choices advised to the player depend only on the current vertex. Such a strategy can be simply described by the set of edges it allows the players to use. $\sigma \subseteq E$ is a positional strategy for player i in the arena G if there is no dead-end in the subgraph $(Dom(\sigma), \sigma)$ induced by σ and σ does not restrict the moves of the adversary: if $v \in V_{1-i} \cap Dom(\sigma)$ then $\{v\} \times vE \subseteq \sigma$. Let $X \subseteq V$ be a subset of vertices. If $Dom(\sigma) = X$, σ is said to be defined on X. We say that a player wins positionally a game Vic on X if he has a positional strategy winning on X.

Subarenas and Traps. Let $X \subseteq V$ be a subset of vertices and $F \subseteq E$ a subset of edges. $G[X]$ denotes the graph $(X, E \cap X^2)$ and $G[X, F]$ denotes the graph $(Dom(F) \cap X, F \cap X^2)$. When $G[X]$ or $G[X, F]$ is an arena, it is said to be a subarena of G. X is said to be a trap for player i in G if $G[X]$ is a subarena and player i can't move outside of X, i.e. $\forall v \in X \cap V_i, vE \subseteq X$.

2.2 Winning Conditions

Let $G = (V, V_0, V_1, E)$ be an arena and $X \subseteq V$. We define various winning conditions.

Attraction game to X. Player 0 wins if the pebble visits X at least once. The corresponding winning condition is $\mathit{Attraction}(X) = V^*XV^\omega$. The winning set for player 0 is denoted by $Att_0(G, X)$ or $Att_0(X)$, when G is obvious. Symmetrically, we define $Att_1(G, X)$ and $Att_1(X)$, the sets of vertices where player 1 can attract the pebble to X. Note that for this game, both players have positional winning strategies.

Trap game and Büchi game to X. Player 0 wins the trap game in X if the pebble stays ultimately in X. The winning condition is $\mathit{Trap}X = V^*X^\omega$. The dual game is the Büchi game to X, where player 0 wins if the pebble visits X infinitely often. The winning condition is $\mathit{Buchi}(X) = (V^*X)^\omega$.

Exploration game. This is a game over an infinite graph, where player 0 wins a play if the pebble visits infinitely many different vertices. The winning condition is $\mathit{Exp} = \{v_0v_1 \cdots \in V^\omega \mid \text{ the set } \{v_0, v_1, \ldots\} \text{ is infinite}\}$.

The exploration condition is an extension of the *Unboundedness* condition introduced in [3]. The Unboundedness condition concerns games played on the configuration graph of a pushdown system. On such a graph, the set of plays is exactly the set of runs of the underlying pushdown automaton, and 0 wins a play if the height of the stack is unbounded, which happens if and only if infinitely many different configurations of the pushdown automaton are visited.

The exploration condition is also closely related to the Σ_3-condition considered in [5], which states that 0 wins a play if every vertex is visited finitely often. Notice that such a play is necessarily also winning for the exploration condition, but the converse is not true. However, given an arena, it is easy to see that each player has the same positional winning strategies for both games. Since the Exploration game is won positionally by both players (cf. Proposition 1), it implies both games have the same winning positions. Hence, in that sense, the Explosion game and the Σ_3-game introduced in [5] are equivalent.

Parity game. G is a parity arena equipped with a priority mapping $\phi : V \to \mathbb{N}$. Player 0 wins a play if there exists a highest priority visited infinitely often and this priority is even, or if the sequence of priorities is unbounded. Thus, the winning condition is

$$Parity_\infty = \{(v_i)_{i\in\mathbb{N}} : \overline{\lim_{i\in\mathbb{N}}}\ \phi(v_i) \in \{0, 2, \ldots, +\infty\}\}$$

where $\overline{\lim}_{i\in\mathbb{N}}\ \phi(v_i) = \lim_{i\in\mathbb{N}}\ \sup_{j\geq i}\ \phi(v_j)$ denotes the limit sup of the infinite sequence of visited priorities. If G is labeled by a finite number of priorities, i.e. if there exists $d \in \mathbb{N}$ such that $\phi : V \to [0, d]$, we write also the winning condition as $Parity_d$. In this case, a classical result [9,19,23] states that both players win this game positionally.

3 Playing the Games $Exp \cup Parity_\infty$ and $Exp \cap Parity_d$

In this section we study the winning strategies for the games $Exp \cup Parity_\infty$ and $Exp \cap Parity_d$. In the case of the game $Exp \cup Parity_\infty$, we show that each player has a positional strategy, winning on the set of his winning posiitons. Concerning the game $Exp \cap Parity_d$, we show that this remains true for player 1, and we exhibit an arena where no winning strategy of player 0 is positional. However, we give a characterization of the winning set of player 0.

3.1 The Game $Exp \cup Parity_\infty$

G is a parity arena equipped with $\phi : V \to \mathbb{N}$.

Proposition 1. *Each player wins positionally the game* $\mathrm{Exp} \cup \mathrm{Parity}_\infty$ *on his winning set.*

Proof. It is crucial to observe that $Exp \cup Parity_\infty$ can be expressed as the limit of a decreasing sequence of winning conditions:

$$Exp \cup Parity_\infty = \bigcap_{n\in\mathbb{N}} Vic_n \ ,$$

where

$$Vic_n = Attraction(\{n+1, n+2, \ldots\}) \cup Parity_\infty \ .$$

Moreover, each game (G, Vic_n) is won positionally by players 0 and 1 on their winning sets X_n and $V \backslash X_n$. It is easy to establish that player 1 wins positionally $(G, \bigcap_n Vic_n)$ on $\bigcup_n V \backslash X_n = V \backslash \bigcap_n X_n$. For winning positionally $\bigcap_n Vic_n$ on

$\bigcap_n X_n$, player 0 can manage to play in such a way that, as long as the pebble stays in $\{0,1,\ldots,n\}$, the play is consistent with a winning strategy for Vic_n. Then, if the pebble stays bounded in some set $\{n, n+1, \ldots\}$, the play is won for condition $\bigcap_{m \geq n} Vic_m \subset Parity_\infty$. If the pebble leaves every set $\{0,\ldots,n\}$, then the play visits infinitely many different vertices and the play is won for *Exp* by player 0. $\square$

Since the *Exp* game is a special case of the $Exp \cup Parity_\infty$ game where all the vertices are labeled with priority 1, we get the following corollary.

Corollary 1. *Each player wins positionally the game* Exp *on his winning set.*

3.2 The Game $Parity_\infty$

A natural question that arises is whether the players have some positional winnign strategies for the $Parity_\infty$ game. Notice that $Exp \subseteq \text{Parity}_\infty$ in the special case where, for every priority d, $\phi^{-1}(d)$ is finite. Indeed, any play visiting infinitely many different vertices will visit infinitely many different priorities.

Hence, in this special case, by Proposition 1, the game Parity_∞ is won positionally by both players. This is not true anymore if $\phi^{-1}(d)$ is infinite for some d. Consider the example given on Fig. 1. The circles are the vertices of player 0 and the squares those of player 1. Player 0 wins Parity_∞ from everywhere but has no positional winning strategy.

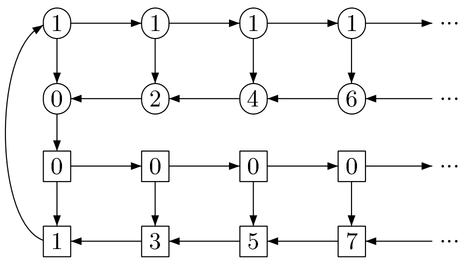

Fig. 1. Player 0's strategy must recall the highest odd vertex reached by player 1 in the lower row in order to answer with a higher even vertex in the second row.

It is interesting to note that if the winning player is determined by the lowest priority visited infinitely often rather than by the greatest one, then both players have positional winning strategies, even if infinitely many priorities are assumed [11].

3.3 The Game $Exp \cap Parity_d$

The analysis of the $Exp \cap Parity_d$ game extends the results of [23]. In this section, G is a parity arena equipped with $\phi : V \to [0, d]$.

Proposition 2. *Player* 1 *wins positionally the game* $\mathrm{Exp} \cap \mathrm{Parity}_d$ *on his winning set.*

Proof. Without loss of generality, we can assume that player 1 wins everywhere. The proof is by induction on d.

If $d = 0$, it is impossible for player 1 to win any play and his winning set is empty. If d is odd and $d \neq 0$, let W be the attractor for player 1 in the set of vertices coloured by the maximal odd priority d. Since $V \backslash W$ is a trap for player 1 coloured from 0 to $d-1$, and by inductive hypothesis, player 1 can win positionally $(G[V \backslash W], \mathit{Exp} \cap \mathit{Parity}_{d-1})$ with some strategy $\sigma_{V \backslash W}$. To win, player 1 shall use $\sigma_{V \backslash W}$ inside $V \backslash W$ and shall attract the pebble to a vertex of colour d when it reaches the set W. That way, either the play stays ultimately in $V \backslash W$ and some suffix is consistent with $\sigma_{V \backslash W}$ or it reaches the odd priority d infinitely often. In both cases, player 1 is the winner.

The case where d is even is less trivial. It is easy to prove that there exists the greatest subarena of G where player 1 wins positionally. It remains to prove that this subarena coincides with the whole arena. □

It may happen that player 0 has a winning strategy from every vertex but he has no positional winning strategy. Such an example is given by Fig. 2.

Fig. 2. To win the $\mathit{Exp} \cap \mathit{Parity}_2$ game, player 0 has to visit new vertices arbitrarily far to the right hand side of the arena and has also to visit the unique vertex of color 2 infinitely often.

Nevertheless, we can characterize the arenas in which player 0 wins the game $\mathit{Exp} \cap \mathit{Parity}_d$ from every position:

Proposition 3. *Let* $G = (V, E)$ *be an arena, coloured from* 0 *to* $d > 0$. *Let* D *be the set of vertices coloured by* d. *Player* 0 *wins the game* $(G, \mathrm{Exp} \cap \mathrm{Parity}_d)$ *on* V *if and only if there exists a subarena* $G[W]$, *coloured from* 0 *to* $d-1$ *such that one of the following conditions holds:*

- **Case d even:** *Player* 0 *wins the games* $(G[W], \mathrm{Exp} \cap \mathrm{Parity}_{d-1})$ *and* (G, Exp) *everywhere and she wins the game* $(G, \mathit{Attraction}(D))$ *on* $V \backslash W$.
- **Case d odd:** *Player* 0 *wins the game* $(G, \mathit{Trap}(W))$ *with a positional strategy* $\sigma_{\mathit{Trap}(W)}$ *and she wins the game* $(G[W, \sigma_{\mathit{Trap}(W)}], \mathrm{Exp} \cap \mathrm{Parity}_{d-1})$.

The conditions of Proposition 3 are illustrated on Fig. 3.

Remark 1. Note that winning the game $(G[W, \sigma_{\mathit{Trap}(W)}], \mathit{Exp} \cap \mathit{Parity}_{d-1})$ means that player 0 has a strategy σ_W winning the game $(G[W], \mathit{Exp} \cap \mathit{Parity}_{d-1})$ which advises player 0 to play moves **consistent with the positional strategy** $\sigma_{\mathbf{Trap(W)}}$.

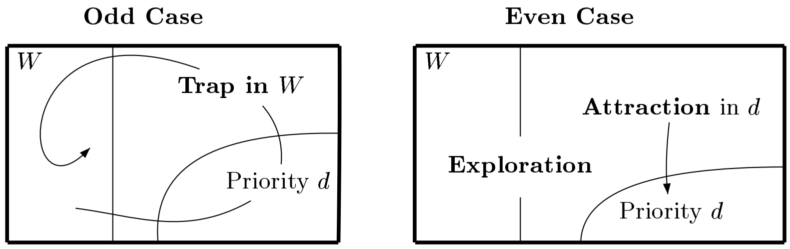

Fig. 3. Conditions of Proposition 3.

Proof. We sketch the proof of the direct implication. In the case where d is even this proof is simple. Consider $W = V\backslash Att_0(D)$. Since $V\backslash W$ is a trap, player 0 wins $(G[V\backslash W], Exp \cap Parity_d)$. The other claims are trivially true.

The case where d is odd is more tricky. We establish first that the family of subarenas of G where Proposition 3 holds is closed by arbitrary union, then we prove that the maximal arena of this family is necessarily G itself.

We sketch the proof of the converse implication. We shall construct a strategy σ_G for player 0 winning the game $(G, Exp \cap Parity_d)$. This construction depends on the parity of d.

d Odd: By hypothesis, player 0 has a positional strategy $\sigma_{Trap(W)}$ winning the game $(G, Trap(W))$ and a strategy σ_{Sub} winning the game $(G[W, \sigma_{Trap(W)}], Exp \cap Parity_{d-1})$. The strategy σ_G is constructed in the following way:

- If the pebble is not in W, player 0 plays according to her positional strategy $\sigma_{Trap(W)}$.
- If the pebble is in W, player 0 uses her strategy σ_{Sub} in the following way: Let p be the sequence of vertices visited up to now and let p' be the longest suffix of p consisting of vertices of W. Player 0 takes a move according to $\sigma_{Sub}(p')$.

The strategy σ_G is winning for the game $(G, Exp \cap Parity_d)$. Indeed, since σ_{Sub} is a strategy in the arena $G[W, \sigma_{Trap(W)}]$, all moves consistent with σ_G are consistent with $\sigma_{Trap(W)}$. Hence, the play is ultimately trapped in W and is ultimately consistent with σ_W, thus won by player 0.

d Even: By hypothesis and by Corollary 1, player 0 has a positional strategy $\sigma_{Exp} \subseteq E$ winning (G, Exp). She has also a positional strategy $\sigma_{Att} \subseteq E$ winning $(G, Attraction(D))$ on $V\backslash W$ and a strategy σ_{Sub} winning the game $(G[W], Exp \cap Parity_{d-1})$.

σ_G is constructed in the following way. At a a given moment player 0 is in one of the three playing modes: *Attraction*, *Sub* or *Exploration*. It can change the mode when the pebble moves to a new vertex. Player 0 begins to play in *Exploration* mode. Here follows the precise description of the strategy σ_G, summarized by Fig. 4.

- The playing mode *Exploration* can occur wherever the pebble is. Player 0 plays according to her positional strategy σ_{Exp}. When a new vertex v is visited for the first time the mode is changed either to *Sub* mode if $v \in W$ or to *Attraction* mode if $v \notin W$.
- The playing mode *Attraction* can occur only if the pebble is not in W. Player 0 plays according to her positional strategy σ_{Att}. When a vertex of priority d is eventually visited, the playing mode is switched to *Exploration*.
- The playing mode *Sub* can occur only if the pebble is in W. Player 0 plays using her strategy σ_{Sub} in the following way. Let p be the sequence of vertices visited up to now and p' the longest suffix of p consisting of vertices of W. Then 0 takes a move according to $\sigma_{Sub}(p')$. If the pebble leaves W, the playing mode is switched to *Exploration*.

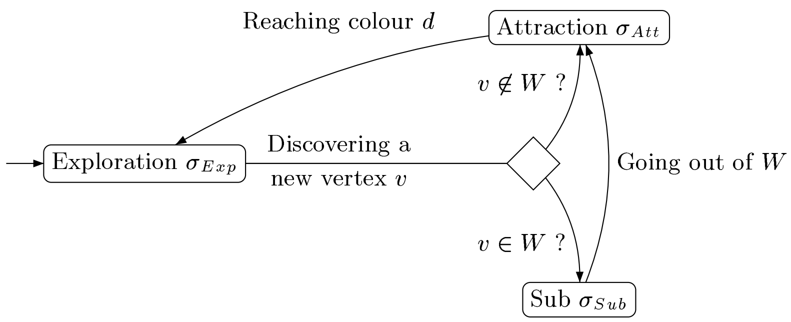

Fig. 4. Rules of transition between playing modes.

Notice that, by definition of σ_{Att} and σ_{Exp}, it is not possible that an infinite play consistent with σ_G stays forever in the playing modes *Attraction* or *Exploration*. Hence, such a play can be of two different types. Either the pebble stays ultimately in the playing mode *Sub* or it goes infinitely often in the modes *Exploration* and *Attraction*. In the first case, it stays ultimately in W and the play is ultimately consistent with σ_{Sub}. In the second case, the pebble visits infinitely often the even priority d and discovers infinitely often a new vertex. In both cases, this play is won by player 0 for the $Exp \cap Parity_d$ condition. □

4 Computation of the Winning Sets and Strategies on Pushdown Arenas

In this section, we apply our results to the case where the infinite graph is the graph of the configurations of a pushdown automaton. And we get an algorithm to compute the winning sets. Moreover, in all cases except for player 0 in the game $Exp \cap Parity_d$, we can also compute winning positional strategies.

Definitions. A pushdown system is a tuple $\mathcal{P} = (Q, \Gamma, \Delta, \perp)$ where Q is a finite set of control states, Γ is a finite stack alphabet, $\perp$ is a special letter called the stack bottom, $\perp \notin \Gamma$ and $\Delta \subseteq Q \times (\Gamma \cup \{\perp\}) \times (\Gamma \cup \{-1\}) \times Q$ is the set of transitions.

The transition $(q, \alpha, \beta, r) \in \Delta$ is said to be a *push transition* if $\beta \in \Gamma$ and a *pop transition* if $\beta = -1$. In both cases, it is said to be an α-transition and a (q, α)-transition. Concerning $\perp$, we impose the restriction that there exists no pop $\perp$-transition. Moreover, we work only with complete pushdown systems, in the sense that, for every couple $(q, \alpha) \in Q \times (\Gamma \cup \{\perp\})$, there exists at least one (q, α)-transition.

Notice that, in the sense of language recognition, any pushdown automaton is equivalent to one of this kind, and the reduction is polynomial.

A configuration of $\mathcal{P}$ is a sequence $q\gamma$, where $q \in Q$ and $\gamma \in \Gamma^*$. Intuitively, q represents the current state of $\mathcal{P}$ while γ is the stack content above the bottom symbol $\perp$. We assume that the symbols on the right of γ are at the top of the stack. Note that $\perp$ is assumed implicitly at the bottom of the stack, i.e. actually the complete stack content is always $\perp\gamma$.

The set of all configurations of $\mathcal{P}$ is denoted by $V_{\mathcal{P}}$. Transition relation $E_{\mathcal{P}}$ over configurations is defined in the usual way: Let $q\gamma\alpha$, where $q \in Q, \gamma \in \Gamma^*$ and $\alpha \in \Gamma$, be a configuration.

- $(q\gamma\alpha, r\gamma) \in E_{\mathcal{P}}$ if there exists a pop transition $(q, \alpha, -1, r) \in \Delta$,
- $(q\gamma\alpha, r\gamma\alpha\beta) \in E_{\mathcal{P}}$ if there exists a push transition $(q, \alpha, \beta, r) \in \Delta$.

Let $q\epsilon$ be a configuration with empty stack. Then

- $(q\epsilon, r\beta) \in E_{\mathcal{P}}$ if there exists a push transition $(q, \perp, \beta, r) \in \Delta$.

We shall write $q\gamma \xrightarrow{\delta} r\gamma'$ to express that a transition $\delta \in \Delta$ of the pushdown automaton corresponds to an edge $(q\gamma, r\gamma') \in E_{\mathcal{P}}$ between two configurations. The graph $G_{\mathcal{P}} = (V_{\mathcal{P}}, E_{\mathcal{P}})$ is called the *pushdown graph* of $\mathcal{P}$.

If Q is partitioned in (Q_0, Q_1), this partition extends naturally to the set of configurations of $\mathcal{P}$ and we $G_{\mathcal{P}}$ is an arena. Moreover, when the control states Q are labeled by priorities with a map $\phi : Q \to [0, d]$, this labeling extends naturally to $V_{\mathcal{P}}$ by setting $\phi(q\gamma) = \phi(q)$. $G_{\mathcal{P}}$ is then a parity arena.

Subgraph Trees and Strategy Trees. With any subset $\sigma \subseteq E_{\mathcal{P}}$ of the edges of a pushdown arena we associate a tree $T_\sigma : \Gamma^* \to 2^{\Delta}$ with vertices labeled by sets of transition of $\mathcal{P}$. This construction is illustrated by Fig. 5.

A vertex of the tree is a stack content of $\mathcal{P}$. A transition $\delta \in \Delta$ belongs to the set labeling a vertex $\gamma \in \Gamma^*$ if there exists a state $q \in Q$ and a configuration $r\gamma'$ such that $q\gamma \xrightarrow{\delta} r\gamma'$ and $(q\gamma, r\gamma') \in \sigma$. Such a tree is called the *coding tree* of σ. Notice that the transformation $\sigma \to T_\sigma$ is one-to-one. If σ is a strategy for player i, we call T_σ a *strategy tree* for player i.

The next theorem states that the languages of positional winning strategies is regular. Thus, we can build a Büchi alternating automaton of size $\mathcal{O}(d|Q|^2 + |\Gamma|)$ which recognizes the language of couples (σ_0, σ_1) such that σ_i is a winning positional strategy for player i and the domains of σ_0 and σ_1 constitute a partition of $V_{\mathcal{P}}$. In the case of the $Parity_d$ and $Exp \cup Parity_d$ games, Proposition 1 establishes

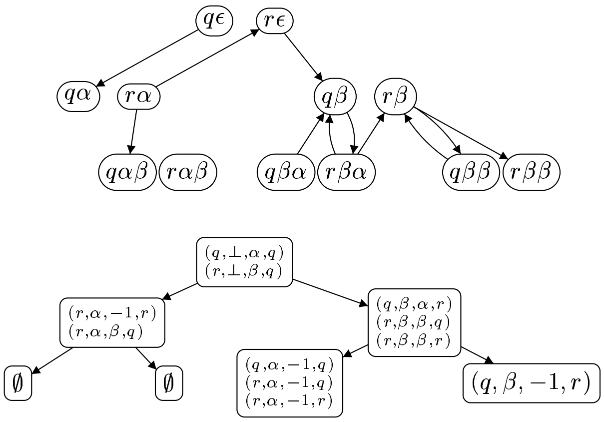

Fig. 5. A finite subset of $E_{\mathcal{P}}$ and its coding tree. Only the labels of the vertices $\{\epsilon,\alpha,\beta,\alpha\alpha,\alpha\beta,\beta\alpha,\beta\beta\}$ are represented. Other vertices of the coding tree are labeled with $\emptyset$.

that this language is non-empty. Hence it is possible to compute a regular tree (σ_0,σ_1) of size $2^{\mathcal{O}(d|Q|^2+|\Gamma|)}$. This regular tree can be seen as the description of a couple of winning stack strategies for both players. This kind of strategy has been defined in [21].

Theorem 1. *Let i be a player and* $\mathrm{Vic} \in \{\mathrm{Parity}_d, \mathrm{Exp}\cup\mathrm{Parity}_d, \mathrm{Exp}\cap\mathrm{Parity}_d\}$. *The language of strategy trees which correspond to winning positional strategies for player i is regular. One can effectively construct an alternating Büchi automaton $\mathcal{A}_{\mathrm{Vic},i}$ with $\mathcal{O}(d|Q|^2+|\Gamma|)$ states which recognizes it.*

Proof. The construction of $\mathcal{A}_{\mathrm{Vic},i}$ uses techniques close to the one of [20,16]. Unfortunately, we couldn't manage to use directly the results of those papers about two-way tree automata, because we don't know how to use a two-way automata to detect a cycle in a strategy tree.

Our aim is to construct a tree automaton recognizing a tree $t:\Gamma^*\to 2^{\Delta}$ iff there exists a winning positional strategy σ such that $t=T_\sigma$. In fact we shall rather construct a Büchi alternating automaton recognizing the complement of the set $\{T_\sigma|\sigma$ winning positional strategy $\}$. First of all it is easy to implement an alternating automaton verifying if the tree t is or is not a strategy tree. It is less trivial to construct the automaton checking if a positional strategy $\sigma\subset E_{\mathcal{P}}$ is winning or not. However, it can be expressed by a simple criterion concerning the cycles and the exploration paths of the graph $(Dom(\sigma),\sigma)$ induced by σ. Those criteria are summarized in Table 1.

We have to construct automata checking each condition of Table 1. They are derived from an automaton detecting the existence of a special kind of finite path called a *jump*. A jump between two vertices with the same stack γ is a path between those vertices, that never pops any letter of γ (see Fig. 4).

Winning Condition	i	Condition on cycles	Condition on exploration paths
Parity	0	Even	Even
	1	Odd	Odd
$Exp \cup Parity_d$	0	Even	No condition
	1	Odd	No exploration path
$Exp \cap Parity_d$	0	No cycle	Even
	1	No condition	Odd

Table 1. Characterization of winning positional strategies.

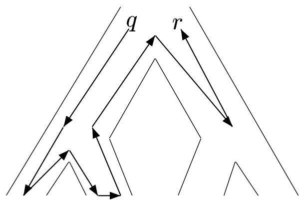

Fig. 6. A jump from $q\gamma$ to $r\gamma$ in a strategy tree.

This kind of path is interesting since a cycle is simply a jump from a vertex to itself, and because the existence of an exploration path of priority c is equivalent to the existence of one of the two kinds of paths illustrated on Fig. 7.

Due to the high computational power of alternation, it is possible to construct automata checking the existence of jumps and detecting the kinds of paths of Fig. 7, with only $\mathcal{O}(d|Q^2| + |\Gamma|)$ control states. $\square$

Computation of winning sets. Using the automata recognizing languages of winning positional strategies, it is possible to recognize the language of winning positions. For each player i, Theorem 2 leads to an EXPTIME procedure to compute a regular tree $\Gamma^* \to 2^Q$ of exponential size that associates with a stack γ the set $\{q \in Q : q\gamma \text{ is winning for player } i\}$. Once computed, deciding if a given position is winning for player i can be done in linear time.

Theorem 2. *For each player i and each winning condition* $\mathrm{Vic} \in \{\mathrm{Parity}_d, \mathrm{Exp} \cup \mathrm{Parity}_d, \mathrm{Exp} \cap \mathrm{Parity}_d\}$*, the tree* $\Gamma^* \to 2^Q$ *which associates with a stack* γ *the set* $\{q \in Q : q\gamma$ *is winning for* $i\}$ *is regular and one can compute a non-deterministic Büchi automaton recognizing it. Such an automaton has* $2^{\mathcal{O}(d|Q|^2+|\Gamma|)}$ *states if* $\mathrm{Vic} \in \{\mathrm{Parity}_d, \mathrm{Exp} \cup \mathrm{Parity}_d\}$ *and* $2^{\mathcal{O}(d^2|Q|^2+d|\Gamma|)}$ *if* $\mathrm{Vic} = \mathrm{Exp} \cap \mathrm{Parity}_d$.

Proof. For the games $Parity_d$ and $Exp \cup Parity_d$, this Theorem is a direct corollary of Theorem 1. In fact, we can build a Büchi alternating automaton which recognizes the language of couples (σ_0, σ_1) such that σ_i is a winning positional strategy for player i and the domains of σ_0 and σ_1 are a partition of $V_{\mathcal{P}}$. The winning sets are then obtained by projection, which requires to transform this alternating automaton to a non-deterministic one and leads to an exponential blowup of the state space [15].

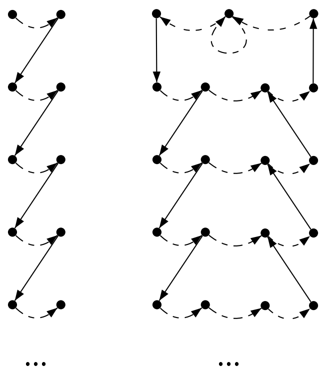

Fig. 7. The dotted arrows are jumps of priority less than c. The top-down regular arrows are push transitions, while the down-top ones are pop-transitions. On the left hand side, infinitely many jumps have priority c. On the right hand side, the upper jump, that is a loop, has priority c.

In the case of the $Exp \cap Parity_d$ game, we use also the characterization of the winning sets given by Proposition 3. We define the notion of a winning-proof, which is a tree on Γ^* labeled by tuples of subsets of Δ, and is defined such that the existence of a winning-proof in an arena is equivalent to the conditions of Proposition 3. Here follows the definition of a winning-proof in a subarena G of a pushdown arena $G_{\mathcal{P}}$.

In the case where $d = 0$, it is a strategy tree $T_{\sigma_{Exp}}$ winning the game (G, Exp).

In the case where $d > 0$ and is even, it is a tuple $T_d = (T', T_{\sigma_{Exp}}, T_{\sigma_{Att}}, T_{d-1})$ where T' is the coding tree of a subarena G' of G, $T_{\sigma_{Exp}}$ is a strategy tree winning the game (G, Exp), $T_{\sigma_{Att}}$ is a strategy tree winning the game $(G, Attraction(D))$ on $Dom(G')$ and T_{d-1} is a $(d-1)$-winning proof in G'.

In the case where d is odd, it is a tuple $T_d = (T', T_{\sigma_{Trap}}, T_{d-1})$ where T' is the coding tree of a subarena G' of G, $T_{\sigma_{Trap}}$ is a strategy tree winning the game $(G, Trap(Dom(G')))$ and T_{d-1} is a $(d-1)$-winning proof in G'.

Each one of those $\mathcal{O}(d)$ conditions can be verified with an alternating automaton with $\mathcal{O}(d|Q|^2 + |\Gamma|)$ states. The corresponding automata constructions are very close to the ones of Theorem 1. Hence, the language of d-winning proofs is regular and recognized by an alternating automaton with $\mathcal{O}(d^2|Q|^2 + d|\Gamma|)$ states.

As in the positional case, by projection, we obtain the desired non-deterministic automaton with $2^{\mathcal{O}(d^2|Q|^2+d|\Gamma|)}$ states. □

Acknowledgements. We thank Wiesław Zielonka and Olivier Serre for some enlightening discussions on games on pushdown graphs, and the anonymous referee for their careful comments.

References

1. A. Arnold, A. Vincent, and I. Walukiewicz. Games for synthesis of controlers with partial observation. *Theoretical Computer Science*, 303(1):7–34, 2003.
2. A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Applications to model checking. In *CONCUR'97, LNCS*, volume 1243, pages 135–150, 1997.
3. A. Bouquet, O. Serre, and I. Walukiewicz. Pushdown games with unboudedness and regular conditions. In *Proc. of FSTTCS'03, LNCS*, volume 2914, pages 88–99, 2003.
4. T. Cachat. Symbolic strategy for games on pushdown graphs. In *Proc. of 29th ICALP, LNCS*, volume 2380, pages 704–715, 2002.
5. T. Cachat, J. Duparc, and W. Thomas. Pushdown games with a σ_3 winning condition. In *Proc. of CSL 2002, LNCS*, volume 2471, pages 322–336, 2002.
6. K. Chatterejee, M. Jurdziński, and T.A. Henzinger. Simple stochastic parity games. In *CSL'03*, volume 2803 of *LNCS*, pages 100–113. Springer, 2003.
7. L. de Alfaro and T.A. Henzinger. Concurrent ω-regular games. In *LICS'00*, pages 142–154. IEEE Computer Society Press, 2000.
8. L. de Alfaro, T.A. Henzinger, and O. Kupferman. Concurrent reachability games. In *FOCS'98*, pages 564–575. IEEE Computer Society Press, 1998.
9. E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. of 32th FOCS*, pages 368–377. IEEE Computer Society Press, 1991.
10. H. Gimbert. Parity and exploration games on infinite graphs, full version. www.liafa.jussieu.fr/~hugo.
11. E. Grädel. Positional determinacy of infinite games. In *STACS 2004, LNCS*, volume 2996, pages 4–18, 2004.
12. O. Kupferman, N. Piterman, and M. Vardi. Pushdown specifictaions. In *Proc. of LPAR'02, LNCS*, October 2002.
13. O. Kupferman and M.Y. Vardi. An automata-theoretic approach to reasoning about infinite state systems. In *Proc. of CAV'00, LNCS*, volume 1855, pages 36–52, 2000.
14. D. E. Muller and P. E. Schupp. The theory of ends, pushdown automata and second order logic. *Thoretical Computer Science*, 37:51–75, 1985.
15. D. E. Muller and P. E. Schupp. Simulating alternating tree automata by nondeterministic automata. *Theoretical Computer Science*, 141:69–107, 1995.
16. N. Piterman and M.Vardi. From bidirectionnality to alternation. *Theoretical Computer Science*, 295:295–321, 2003.
17. O. Serre. Games with winning conditions of high borel complexity, to appear in proc. of icalp'04.
18. O. Serre. Note on winning positions on pushdown games with omega-regular conditions. *Information Processing Letters*, 85(6):285–291, 2003.
19. W. Thomas. On the synthesis of strategies in infinite games. In *Proc. of STACS'95,LNCS*, volume 900, pages 1–13, 1995.
20. M. Vardi. Reasoning about the past with two-way automata. In *Proc. of ICALP'98, LNCS*, volume 1443, pages 628–641, 1998.
21. I. Walukiewicz. Pushdown processes: Games and model checking. In *Proc. of CAV'96, LNCS*, volume 1102, pages 62–74, 1996.
22. I. Walukiewicz. Pushdown processes: Games and model checking. *Information and Computation*, 164(2):234–263, 2001.
23. W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.