A Dijkstra-type algorithm for dynamic games

Martino Bardi, Juan Pablo Maldonado Lopez

A Dijkstra-type algorithm for dynamic games

Martino Bardi*
Università degli Studi di Padova
Dipartimento di Matematica
Via Trieste 63-35121
Padova, Italy.
bardi@math.unipd.it

Juan Pablo Maldonado Lopez†
Université Pierre et Marie Curie
Equipe Combinatoire et Optimisation
4 Place Jussieu - 75252
Paris, France
maldonadolo@math.jussieu.fr

August 9, 2013

Abstract

We study zero-sum dynamic games with deterministic transitions where player 1 knows player 2's move, as well as games where players make simultaneous moves and the transitions are stochastic and depending on their actions and the state. Player 1 aims at reaching a terminal set and minimising a running and final cost. We propose and analyse an algorithm that computes the value function of these games extending Dijkstra's algorithm for shortest paths on graphs.

1 Introduction

In this paper we study zero-sum dynamic games where players move a state variable in a state space $\mathcal{X}$ and they incur in a discounted cost that depend on their positions and actions. Additionally, player 1 aims at reaching a given terminal set $\mathcal{X}_f$ and once this is done the game is finished and a final cost is assigned. These games are special cases of general zero-sum stochastic games as introduced by Shapley [21] and are also related to recursive games [12]. In those papers the existence of the discounted value function is obtained, among other results. Our assumptions are designed to cover discrete approximations of generalised pursuit-evasion differential games [6, 4, 3], see also the survey [5] and the references therein. Related problems are the *reachability games* recently studied in [1], where also some numerical algorithms are provided. Our purpose is to provide an efficient algorithm to compute the value.

Several algorithms have been proposed to compute the value function of stochastic games in the finite case (finite state space and finite action sets). The most classical, going back to Shapley, is the *value iteration*: in the seminal paper [21], a recursive formula is established and since the value of the discounted dynamic game is the unique fixed point of a contractive operator, its iterates converge. Several variants of this idea have been proposed to accelerate convergence, see for instance the survey [13], the more recent paper [18] and the references therein, and [16] where a Gauss-Seidel procedure for value iteration is studied.

Our approach, instead, is inspired by the Dijkstra algorithm [11] for finding shortest paths in finite graphs, which has running time $O(e + v \log v)$ if a suitable data structure is used, where v, e denote respectively the number of vertices and edges. We refer to [14] for the details. In contrast to value iteration, the algorithm we propose updates the approximate value function only in the immediate neighbors of those nodes where the value is already computed, thus reducing the computation time, and converges in a finite number of steps.

*Partially supported by the Fondazione CaRiPaRo Project "Nonlinear Partial Differential Equations: models, analysis, and control-theoretic problems", the MIUR project PRIN "Viscosity, geometric, and control methods for nonlinear diffusive models", and the European Project Marie Curie ITN "SADCO - Sensitivity Analysis for Deterministic Controller Design".

†Partially supported by the Commission of the European Communities under the 7th Framework Programme Marie Curie Initial Training Network (FP7-PEOPLE-2010-ITN), project SADCO, contract number 264735.

Our first motivation comes from the so-called *Fast Marching Methods* (briefly, FMM) for Hamilton-Jacobi equations with convex Hamiltonian arising in deterministic control and front propagation problems, introduced in [23, 19] and developed in [20], see also the references therein. These numerical methods, also called *single-pass*, approximate time-optimal control in continuous time and space with a fully discrete optimization problem on a grid, and then rely on the classical Dijkstra's algorithm for an efficient solution of the discrete approximation. We recall that the methods based on Dynamic Programming have several good properties, especially robustness, but they face the well-known "curse of dimensionality". A large amount of research in the last twenty years were devoted to overcoming this difficulty in some cases and FMM played an important role for problems with positive costs (see also [17, 9] and the references therein for other approaches).

Recently various forms of FMM were also used for solving some Hamilton-Jacobi-Isaacs equations arising from differential games [15, 10, 8], with possible applications to the stabilization of perturbed systems and to front propagation problems with non-convex Hamiltonian. They approximate the continuous problem with a discrete-time game on a grid. However, so far there is no theoretical justification for using Dijkstra-type algorithms for such discrete dynamic games. One of the goals of this paper is providing a rigorous foundation to these methods.

2 The model

Let $\mathcal{X}$ be a finite set belonging to an euclidean space $\mathbb{R}^d$, which we call the *state space*. Let A, B be finite sets where the players choose their controls. For a function $S : \mathcal{X} \times A \times B \to \mathcal{X}$ define the trajectory $x_\bullet = x_\bullet(x, a_\bullet, b_\bullet)$ recursively by

$$x_{n+1} = S(x_n, a_n, b_n), \;\; x_0 = x. \tag{1}$$

Let $\mathcal{X}_f \subset \mathcal{X}$, denote a *terminal set* of nodes (which player 1 wishes to attain) and let $\gamma \in (0, 1]$ be a discount factor. We introduce the *running* and *terminal* cost

$$\ell : \mathcal{X} \times A \times B \to \;\; \mathbb{R}, \quad 0 < \ell_0 \le \ell(x, a, b) \le L, \, \forall (x, a, b) \in \mathcal{X} \times A \times B \tag{2}$$

$$g : \mathcal{X}_f \to \;\; \mathbb{R}, \quad g_0 \le g(x) \le g_1, \forall x \in \mathcal{X}_f \tag{3}$$

and additionally define the *arrival time* $\hat{n} : \mathcal{X} \times A^{\mathbb{N}} \times B^{\mathbb{N}} \to \mathbb{R}$ by

$$\hat{n}(x, a_\bullet, b_\bullet) = \begin{cases} \min\{n \in \mathbb{N} : x_n \in \mathcal{X}_f\}, & \text{if } \{n \in \mathbb{N} : x_n \in \mathcal{X}_f\} \neq \emptyset \\ +\infty & \text{else,} \end{cases}$$

where x_n is the trajectory of (1) corresponding to the control sequences $a_\bullet, b_\bullet$. To alleviate the notation, we will often write $\hat{n}$ instead of the more explicit $\hat{n}(x, a_\bullet, b_\bullet)$ when no confusion arises. We have then the following *total cost* functional $J : \mathcal{X} \times A^{\mathbb{N}} \times B^{\mathbb{N}} \to \mathbb{R}$

$$J(x, a_\bullet, b_\bullet) := \sum_{n=0}^{\hat{n}-1} \ell(x_n, a_n, b_n)\gamma^n + \gamma^{\hat{n}} g(x_{\hat{n}}).$$

Observe that the if $\hat{n} = +\infty$ the cost is finite for $\gamma < 1$ and $+\infty$ for $\gamma = 1$. Player 1 chooses $a_\bullet \in A^{\mathbb{N}}$ and player 2 chooses $b_\bullet \in B^{\mathbb{N}}$. The aim of player 1 is to minimize the cost functional, whereas player 2 has the opposite goal.

We assume that both players observe each other's actions and the state $x_\bullet$. We refer to $\mathcal{G} = \mathcal{G}\langle \mathcal{X}, \mathcal{X}_f, S, A, B, \ell, g, \gamma \rangle$ as *the game*. We will consider two modes of play: alternating and simultaneous moves.

3 Alternating moves

3.1 The lower value function

We consider in this section the case when player 1 knows the action that player 2 will play at each time. This is relevant, for instance, in the discretization of the lower value of a differential game, or in the case

of discrete robust control problems, where player 2 represents a disturbance.

Definition 1. A map $\alpha : B^{\mathbb{N}} \to A^{\mathbb{N}}$ is a *non anticipating strategy* for player 1 if

$$b_n = \tilde{b}_n, \forall n \leq m \implies \alpha[b_\bullet]_n = \alpha[\tilde{b}_\bullet]_n, \forall n \leq m.$$

Denote with $\mathcal{A}$ the set of non anticipating strategies for player 1. The definition of the set $\mathcal{B}$ of non anticipating strategies for player 2 is completely analogous.

This allows us to introduce the *lower value function*

$$V^-(x) := \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} J(x, \alpha[b_\bullet], b_\bullet).$$

Note that the infimum and supremum in this definition are attained since the action and state spaces are finite. The following result follows from familiar arguments, see for instance [2, Chapter 8, Theorem 3.18].

Proposition 2. *The lower value function satisfies*

$$V^-(x) = \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} \left\{ \sum_{n=0}^{k \wedge \hat{n} - 1} \ell(x_n, \alpha[b_\bullet]_n, b_n) \gamma^n + \gamma^{k \wedge \hat{n}} V^-(x_{k \wedge \hat{n}}) \right\}, \forall k \in \mathbb{N}. \tag{4}$$

$$V^-(x) = \max_{b \in B} \min_{a \in A} \left\{ \ell(x, a, b) + \gamma V^-(S(x, a, b)) \right\}, \ \forall x \notin \mathcal{X}_f \tag{5}$$

$$V^-(x) = g(x), \ \forall x \in \mathcal{X}_f. \tag{6}$$

The first equality (4) is the well known dynamic programming property. By taking $k = 1$ in (4) one can easily prove (5). The last equality (6) follows directly from the definition.

The following form of the dynamic programming property will be useful later.

Proposition 3. *Let $\mathcal{X}_f \subset \tilde{\mathcal{X}} \subset \mathcal{X}$ and let $\tilde{n}$ denote the arrival time to $\tilde{\mathcal{X}}$, i.e. $\tilde{n} = \tilde{n}(x, a_\bullet, b_\bullet) = \inf\{n \in \mathbb{N} : x_n \in \tilde{\mathcal{X}}\}$. Then*

$$V^-(x) = \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} \left\{ \sum_{n=0}^{\tilde{n} - 1} \ell(x_n, \alpha[b_\bullet]_n, b_n) \gamma^n + \gamma^{\tilde{n}} V^-(x_{\tilde{n}}) \right\}.$$

Proof. This is a direct consequence of the dynamic programming property (4) since $\tilde{n} \leq \hat{n}$. □

3.2 The algorithm

The following algorithm computes the value function:

Require: $n = 0, \mathrm{Acc}_0 := \mathcal{X}_f$, $W_0(x) := +\infty, \forall x \in \mathcal{X}$, $V_0^-(x) = g(x), \forall x \in \mathcal{X}_f$
  while $\mathrm{Cons}_n \neq \emptyset$ **do**
    $A_n(x) := \{a \in A : S(x, a, b) \in \mathrm{Acc}_n, \forall b \in B\}$
    $\mathrm{Cons}_n := \{x \in \mathcal{X} \setminus \mathrm{Acc}_n : A_n(x) \neq \emptyset\}$
    $W_{n+1}(x) := \max_{b \in B} \min_{a \in A_n(x)} \{\ell(x, a, b) + \gamma V_n^-(S(x, a, b))\}, \forall x \in \mathrm{Cons}_n$
    $\mathrm{Acc}_{n+1} := \mathrm{Acc}_n \cup \mathrm{argmin} W_{n+1}$
    $V_{n+1}^-(x) := W_{n+1}(x), \forall x \in \mathrm{argmin} W_{n+1}$
    $V_{n+1}^-(x) := V_n^-(x), \forall x \in \mathrm{Acc}_n$
    $n \leftarrow n + 1$
  end while

The notations introduced in the algorithm have the following meaning

- Acc_n is the set of nodes accepted at the step n, at such nodes the approximate value is not re-computed in the next steps;
- $A_n(x) \subseteq A$ is the set of controls that take the state x to Acc_n, no matter what player 2 does;

- Cons_n is the set of nodes considered at the step n, i.e., those form which player 1 can reach Acc_n;
- $x \in \text{argmin} W_{n+1}$ if $W_{n+1}(x) = \min_{\mathcal{X}} W_{n+1}$, such nodes become accepted at step $n+1$.

Note that Acc_n is strictly increasing as long as $\text{Cons}_n \neq \emptyset$, so the algorithm terminates in a finite number N of steps, at most the cardinality of $\mathcal{X} \setminus \mathcal{X}_f$, which we denote with $|\mathcal{X} \setminus \mathcal{X}_f|$. Denote also with $\mathcal{R}$ the set of nodes from which player 1 can reach the terminal set for any behavior of player 2, i.e.,

$$\mathcal{R} := \{x \in \mathcal{X} : \inf_{\alpha \in \mathcal{A}} \sup_{b_\bullet \in B^{\mathbb{N}}} \hat{n}(x, \alpha[b_\bullet], b_\bullet) < +\infty\}.$$

It is easy to see that if N is the terminal step of the algorithm, i.e., $\text{Cons}_N = \emptyset$, then $\text{Acc}_N = \mathcal{R}$.

The main result of this section states that the algorithm indeed computes the value function. It requires the following additional assumption in the discounted case $\gamma < 1$ (see Remark 8 below for a discussion about this condition).

Condition 4. If $\gamma < 1$

$$L + \gamma g_1 \leq \frac{\ell_0}{1-\gamma}.$$

Proposition 5. *Assume either $\gamma = 1$ or $\gamma < 1$ and Condition 4. Then, for any $n \leq N$,*

$$V_n^-(x) = V^-(x), \quad \textit{for all } x \in \text{Acc}_n,$$

and the algorithm converges in $N \leq |\mathcal{X} \setminus \mathcal{X}_f|$ steps to the value function V^- on the reachability set $\mathcal{R}$.

Proof. Observe that for $n = 0$ the conclusion holds by definition. It suffices to prove that $V_1^-(x) = V^-(x)$ for $x \in \text{Acc}_1$ since by Proposition (3), if we know V^- on $\tilde{\mathcal{X}} = \text{Acc}_1$, then we can obtain V^- as the value of the new problem with $\mathcal{X}_f$ replaced by $\tilde{\mathcal{X}}$ and g by $V^-|_{\tilde{\mathcal{X}}}$ and thus conclude by induction.

Observe first that $V_1^-(x) \geq V^-(x)$ follows easily from the definitions. Now for

$$\bar{x} \in \text{argmin}_{x \in \text{Cons}_1} W_1(x)$$

consider an optimal pair $(\alpha^*, b_\bullet^*) \in \mathcal{A} \times B^{\mathbb{N}}$ and the corresponding optimal trajectory x_n starting from $\bar{x}$, that is,

$$x_{n+1} = S(x_n, \alpha^*[b_\bullet^*]_n, b_n^*), \quad x_0 = \bar{x},$$
$$V^-(\bar{x}) = J(\bar{x}, \alpha^*[b_\bullet^*], b_\bullet^*).$$

If $\hat{n}(\bar{x}, \alpha^*[b_\bullet^*], b_\bullet^*) = 1$ then $V^-(\bar{x}) = W_1(\bar{x}) = V_1^-(\bar{x})$, which is the desired conclusion. If, instead, $\hat{n} := \hat{n}(\bar{x}, \alpha^*[b_\bullet^*], b_\bullet^*) > 1$ we will distinguish two cases.

- Case $\gamma = 1$. From (4) and $\ell > 0$ we have that

$$V^-(\bar{x}) = \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*) + V^-(x_{\hat{n}-1}) > V^-(x_{\hat{n}-1}).$$

  On the other hand, we have an optimal pair strategy-control and corresponding optimal trajectory starting from $x_{\hat{n}-1}$ that reaches $\mathcal{X}_f$ in one step. Then $V^-(x_{\hat{n}-1}) = W_1(x_{\hat{n}-1})$ and so

$$V^-(x_{\hat{n}-1}) = W_1(x_{\hat{n}-1}) \geq W_1(\bar{x}) = V_1^-(\bar{x}) \geq V^-(\bar{x})$$

  which is a contradiction.

- Case $\gamma < 1$. Here (4) gives

$$\begin{aligned} V^-(\bar{x}) &= \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*)\gamma^n + \gamma^{\hat{n}-1} V^-(x_{\hat{n}-1}) \\ &= \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*)\gamma^n + \gamma^{\hat{n}-1} W_1(x_{\hat{n}-1}) \\ &\geq \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*)\gamma^n + \gamma^{\hat{n}-1} W_1(\bar{x}) \\ &\geq \sum_{n=0}^{\hat{n}-2} \ell(x_n, \alpha^*[b_\bullet]_n, b_n^*)\gamma^n + \gamma^{\hat{n}-1} V^-(\bar{x}). \end{aligned}$$

  Then,

$$V^-(\bar{x})(1-\gamma^{\hat{n}-1}) \geq \ell_0 \frac{1-\gamma^{\hat{n}-1}}{1-\gamma} \implies V^-(\bar{x}) \geq \frac{\ell_0}{1-\gamma}.$$

  On the other hand, since

$$V_1^-(\bar{x}) \leq L + \gamma g_1$$

  we get the inequality $V_1^-(\bar{x}) \leq V^-(\bar{x})$ by Condition 4.

□

Remark 6. The main advantage of our algorithm is that at each step the approximate value function is updated only on some nodes, namely on Cons_n. Moreover, at least one of these nodes becomes accepted and will not be considered in the next steps. In particular, if the costs l and g are constant (generalized pursuit-evasion games), then W_{n+1} is constant on Cons_n and all considered nodes are accepted. In other words, the value function is computed only once on each node, which considerably speeds up the algorithm in comparison to iterative methods. Algorithms with this property are often called *single-pass*.

Remark 7. If we stop the algorithm before it terminates, say at $\tilde{n} < N$, Proposition 5 says that we have anyway computed the value function in the set $\mathrm{Acc}_{\tilde{n}}$, which may be enough for some practical problems with very large grids.

Remark 8. Condition 4 requires that the oscillation of the running cost ℓ and the size of the terminal cost g are not too high compared with $1/(1-\gamma)$. It essentially says that, if player 1 is on a node where he can reach $\mathcal{X}_f$, it is convenient for him to do so even if the cost of this step is high, rather than following forever a trajectory with cheap running costs. For any ℓ and g verifying (2) and (3) the condition is satisfied for γ sufficiently close to 1. It is satisfied also for all $\gamma \in (0,1)$ in discounted pursuit-evasion games, where $\ell \equiv 1$ and $g \equiv 0$.

Remark 9. If $\gamma = 1$, we can add a final step to the algorithm by setting $V_{N+1}^-(x) := W_0(x) = +\infty$ for all $x \in \mathcal{X} \setminus \mathrm{Acc}_N$, so $V_{N+1}^-(x) = V^-(x)$ for $x \in \mathcal{X} \setminus \mathcal{R}$ and we have convergence on the whole state space $\mathcal{X}$. On the other hand, if $\gamma < 1$ the algorithm gives no information on the value V^- outside $\mathcal{R}$.

Remark 10. If $\gamma = 1$, $\ell \equiv 1$, and $g \equiv 0$, the problem for player 1 is the shortest length of paths reaching $\mathcal{X}_f$, whereas player 2 seeks to make such length the longest possible (generalised pursuit-evasion). If, in addition, there is no player 2, i.e., B is a singleton, the problem reduces to the shortest path and the algorithm of this section is the classical Dijkstra algorithm.

In reachability games there is no running cost and all the states in $\mathcal{X}_f$ have the same cost. Then the algorithm is essentially the same as Algorithm 2 in [1], where the set Acc_{n+1} is updated by

$$\mathrm{Acc}_{n+1} := \mathrm{Acc}_n \cup \mathrm{Cons}_n.$$

Then the reachability set $\mathcal{R}$ is computed exactly as in the above generalised pursuit-evasion game, although here the length of the path is not of interest. When the moves are alternating, this algorithm runs in a linear time with respect to the size of the game, defined as [1]

$$\|\mathcal{G}\| := |X| + |A| + |B|.$$

[1] This definition is slightly different than the one in [1], since there only the actions available at each state are considered. Without loss of generality, we assume that all actions are available in all states.

4 Simultaneous moves

In the case of simultaneous moves, players in general will need to select their actions randomly. The description of the strategy sets is a bit involved in general since it requires to define appropriately the information available to the players, see [22] for instance for the details. However, it turns out that [21] it suffices to consider stationary strategies, i.e. functions of the form $\sigma : \mathcal{X} \to \Delta(A)$ for player 1, where $\Delta(A)$ denotes the set of probability distributions in A. Analogously, for player 2 the stationary strategies are functions of the form $\tau : \mathcal{X} \to \Delta(B)$. The sets of stationary strategies are denoted by Σ and $\mathcal{T}$ for players 1 and 2 respectively.

Let $(\sigma, \tau) \in \Sigma \times \mathcal{T}$. Together with the (not necessarily deterministic) transition S, they define a discrete-time Markov chain $(X_n^{\sigma\tau})_{n\in\mathbb{N}}$ with state space $\mathcal{X}$ and transition probability induced by the strategies σ and τ and denoted $\mathbb{P}_{\sigma\tau}$. Let $\hat{n}_{\sigma\tau}$ denote the arrival time of the Markov chain $(X_n^{\sigma\tau})_{n\in\mathbb{N}}$ to $\mathcal{X}_f$ starting from x, i.e.

$$\hat{n}_{\sigma\tau}(x) := \inf\{n \in \mathbb{N} \ : \ X_n^{\sigma\tau} \in \mathcal{X}_f \mid X_0 = x\}.$$

The cost functional is then

$$J(x, \sigma, \tau) := \mathbb{E}_{\sigma\tau}\left[\sum_{n=0}^{\hat{n}_{\sigma\tau}-1} \ell(x_n, a_n, b_n)\gamma^n + \gamma^{\hat{n}_{\sigma\tau}} g(x_{\hat{n}_{\sigma\tau}})\right].$$

Consider

$$s_x = \begin{cases} 1, & \text{if } \ x \in \mathcal{X}_f \\ 0 & \text{else.} \end{cases}$$

and consider a new transition function $\tilde{S} : \mathcal{X} \times A \times B \to \mathcal{X}$ given by

$$x_{n+1} = \tilde{S}(x_n, a_n, b_n) := s_{x_n} \cdot x_n + (1 - s_{x_n}) \cdot S(x_n, a_n, b_n)$$

which means that once a state in $\mathcal{X}_f$ is reached, the state variable does not move. Hence we can rewrite the cost functional as

$$J(x, \sigma, \tau) = \mathbb{E}_{\sigma\tau}\left[\sum_{n=0}^{+\infty}\left\{(1 - s_{x_n}) \cdot \ell(x_n, a_n, b_n) + s_{x_n} \cdot \frac{1}{1-\gamma} \cdot g(x_n)\right\} \cdot \gamma^n\right].$$

This motivates us to introduce the following cost function

$$\tilde{\ell}(x, a, b) := (1 - s_x) \cdot \ell(x, a, b) + s_x \cdot \frac{1}{1-\gamma} \cdot g(x).$$

This slightly modified game with transition $\tilde{S}$ and cost function $\tilde{\ell}$ is equivalent in terms of costs and strategies to the original game. When we write it in this form, it is easy to see that this game is a special case of discounted stochastic games with finite state and finite action spaces, as studied by Shapley [21] and thus it admits a value, i.e.

$$\min_{\sigma\in\Sigma}\max_{\tau\in\mathcal{T}} J(x, \sigma, \tau) = \max_{\tau\in\mathcal{T}}\min_{\sigma\in\Sigma} J(x, \sigma, \tau) \tag{7}$$

and their common value is denoted $V(x)$. Moreover, the following dynamic programming property holds:

$$V(x) = \max_{\beta\in\Delta(B)}\min_{\alpha\in\Delta(A)} \mathbb{E}_{\alpha\beta}\left\{\tilde{\ell}(x, a, b) + \gamma V(S(x, a, b))\right\},$$

i.e., the value function is a fixed point of Shapley's map.

In the case of simultaneous moves, we propose the following algorithm, which is very similar to Algorithm 3.2:
Also this algorithm terminates in a finite number of steps, that we denote again with N. The main result of this section is the following convergence theorem.

Require: $n = 0, \mathrm{Acc}_0 := \mathcal{X}_f$, $W_0(x) := +\infty, \forall x \in \mathcal{X}$, $V_0(x) = g(x), \forall x \in \mathcal{X}_f$
  while $\mathrm{Cons}_n \neq \emptyset$ **do**
    $A_n(x) := \{a \in A : S(x,a,b) \in \mathrm{Acc}_n, \forall b \in B\}$
    $\mathrm{Cons}_n := \{x \in \mathcal{X} : A_n(x) \neq \emptyset\}$
    $W_{n+1}(x) := \max_{b \in \Delta(B)} \min_{a \in \Delta(A_n(x))} \mathbb{E}_{\alpha\beta}\{\ell(x,a,b) + \gamma V_n(S(x,a,b))\}, \forall x \in \mathrm{Cons}_n$
    $\mathrm{Acc}_{n+1} := \mathrm{Acc}_n \cup \mathrm{argmin} W_{n+1}$
    $V_{n+1}(x) := W_{n+1}(x), \forall x \in \mathrm{argmin} W_{n+1}$
    $V_{n+1}(x) := V_n(x), \forall x \in \mathrm{Acc}_n$
    $n \leftarrow n+1$
  end while

Proposition 11. *Assume either $\gamma = 1$ or $\gamma < 1$ and Condition 4. Then, for any $n \leq N$,*

$$V_n(x) = V(x), \quad \textit{for all } x \in \mathrm{Acc}_n,$$

and the algorithm converges in $N \leq |\mathcal{X} \setminus \mathcal{X}_f|$ steps to the value function V^- on the set Acc_N.

Proof. The proof mimics that of Proposition 5. We show it only on the case $\gamma < 1$. From the arguments in Proposition 5, it remains to show $V_1 \leq V$. Let σ^*, τ^* optimal strategies for the players and let $\bar{x} \in \mathrm{argmin}_{\mathrm{Cons}_1} W_1$. If $\mathrm{supp}\sigma^*[\bar{x}] \subseteq A_1(\bar{x})$, then $\hat{n}_{\sigma^*\tau^*}(\bar{x}) \equiv 1$ and the conclusion is achieved. If this is not the case, then there exists an action a^* such that $\sigma^*[\bar{x}](a^*) > 0$ and such that there exists b^* with $S(\bar{x}, a^*, b^*) \notin \mathcal{X}_f$. If player 1 uses the strategy σ^* and player 2 plays a strategy τ_1 consisting of playing b^* at the initial position $\bar{x}$ and arbitrarily afterwards, then $\mathbb{P}(\hat{n}_{\sigma^*\tau_1} > 1) > 0$. We have that

$$\begin{aligned} V(\bar{x}) &= \max_{\tau \in \mathcal{T}} \mathbb{E}_{\sigma^*\tau}\left[\sum_{n=0}^{+\infty} \tilde{\ell}(x_n, a_n, b_n)\gamma^n\right] \\ &\geq \mathbb{P}(\hat{n}_{\sigma^*\tau_1} > 1) \cdot \mathbb{E}_{\sigma^*\tau_1}\left[\sum_{n=0}^{\hat{n}-2} \ell(x_n, a_n, b_n)\gamma^n + \gamma^{\hat{n}-1} V(x_{\hat{n}-1}) \mid \hat{n}_{\sigma^*\tau_1} > 1\right] \\ &\geq \mathbb{P}(\hat{n}_{\sigma^*\tau_1} > 1) \cdot \mathbb{E}_{\sigma^*\tau_1}\left[\sum_{n=0}^{\hat{n}-2} \ell_0\gamma^n + \gamma^{\hat{n}-1} V(\bar{x}) \mid \hat{n}_{\sigma^*\tau_1} > 1\right]. \end{aligned}$$

Altogether this implies that

$$0 \geq \mathbb{P}(\hat{n}_{\sigma^*\tau_1} > 1) \cdot \mathbb{E}_{\sigma^*\tau_1}\left[1 - \gamma^{\hat{n}_{\sigma^*\tau_1}-1} \mid \hat{n}_{\sigma^*\tau_1} > 1\right] \cdot \left(\frac{\ell_0}{1-\gamma} - V(\bar{x})\right)$$

from which we obtain $V(\bar{x}) \geq \frac{\ell_0}{1-\gamma}$. We conclude as in Proposition 5. □

Remark 12. The adaptation of the algorithm to the case of stochastic transitions presents several difficulties. For a single player the Dijkstra algorithm was extended to the stochastic case in [7] if there exists a consistently improving optimal policy. A deeper study of the causality properties needed in stochastic shortest path problems with one controller is in [24]. The case of 0-sum two-person games appears to be open.

References

[1] L. Alfaro, T. Henzinger, and O. Kupferman, *Concurrent reachability games*, Theoretical Computer Science **xx** (2007), no. 386, 188–217.

[2] M. Bardi and I. Capuzzo-Dolcetta, *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*, Systems & Control: Foundations & Applications, Birkhäuser Boston Inc., Boston, MA, 1997, With appendices by Maurizio Falcone and Pierpaolo Soravia.

[3] Bardi, M.; Bottacin, S.; Falcone, M.: Convergence of discrete schemes for discontinuous value functions of pursuit-evasion games. New trends in dynamic games and applications, 273–304, Ann. Internat. Soc. Dynam. Games, 3, Birkhäuser Boston, Boston, MA, 1995.

[4] Bardi, M.; Falcone, M.; Soravia, P.: Fully discrete schemes for the value function of pursuit-evasion games. Advances in dynamic games and applications (Geneva, 1992), 89–105, Ann. Internat. Soc. Dynam. Games, 1, Birkhäuser Boston, Boston, MA, 1994.

[5] Bardi, M.; M. Falcone; P. Soravia: Numerical methods for pursuit-evasion games via viscosity solutions, in "Stochastic and differential games: theory and numerical methods", M. Bardi, T. Parthasarathy e T.E.S. Raghavan eds., pp. 105-175, Ann. Internat. Soc. Dynam. Games, 4, Birkhäuser, Boston, 1999.

[6] Bardi, M.; Soravia, P.: Approximation of differential games of pursuit-evasion by discrete-time games. Differential games—developments in modelling and computation (Espoo, 1990), 131–143, Lecture Notes in Control and Inform. Sci., 156, Springer, Berlin, 1991.

[7] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed., Vol. II. Athena Scientific, Boston, 2001.

[8] S. Cacace, E. Cristiani, and M. Falcone, *A local ordered upwind method for Hamilton-Jacobi and Isaacs equations*, 18th IFAC World Congress 18, 2012.

[9] S. Cacace, E. Cristiani, M. Falcone and A.Picarelli, *A patchy dynamic programming scheme for a class of Hamilton-Jacobi-Bellman equations*. SIAM J. Sci. Comput. 34 (2012), no. 5.

[10] E. Cristiani, *A fast marching method for Hamilton-Jacobi equations modeling monotone front propagations*, J. Sci. Comput. **39** (2009), 189–205.

[11] E.W. Dijkstra, *A note on two problems in connection with graphs*, Numer. Math., 1 **1** (1959), 269–271.

[12] H. Everett, *Recursive games*, Contributions to the theory of games, vol. 3, Annals of Mathematics Studies, no. 39, Princeton University Press, Princeton, N. J., 1957, pp. 47–78.

[13] J.A. Filar and T. E. S. Raghavan, *Algorithms for stochastic games: A survey*, Zeischrift fuer Operations Research **35** (1991), no. 6, 437–472.

[14] M.L. Fredman and R. E. Tarjan, *Fibonacci heaps and their uses in improved network optimization algorithms*, 25th Annual Symposium on Foundations of Computer Science (IEEE) **xx** (1984), 338–346.

[15] L. Grüne and O. Junge, *Global optimal control of perturbed systems*. J. Optim. Theory Appl. 136 (2008), no. 3, 411–429.

[16] H.J. Kushner, *The Gauss-Seidel numerical procedure for Markov stochastic games*, IEEE Transactions on Automatic Control **49** (2004), no. 10, 1779 – 1784.

[17] W.M. McEneaney, *Max-plus methods for nonlinear control and estimation*. Birkhäuser Boston, Inc., Boston, MA, 2006.

[18] T. E. S. Raghavan and Z. Syed, *A policy-improvement type algorithm for solving zero-sum two-person stochastic games of perfect information*. Math. Program. 95 (2003), no. 3, Ser. A, 513–532.

[19] J.A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Nat. Acad. Sci. U.S.A. **93** (1996), 1591–1595.

[20] J.A. Sethian, *Level set methods and fast marching methods. Evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Second edition.* Cambridge Monographs on Applied and Computational Mathematics, 3. Cambridge University Press, Cambridge, 1999

[21] L. S. Shapley, *Stochastic games*, Proc. Nat. Acad. Sci. U. S. A. **39** (1953), 1095–1100.

[22] S. Sorin, *A first course on zero-sum repeated games*, Mathématiques & Applications vol. 37, Springer-Verlag, Berlin, 2002.

[23] J.N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Transactions on Automatic Control **40** (1995), 1528–1538.

[24] A. Vladimirsky, *Label-setting methods for multimode stochastic shortest path problems on graphs*, Math. Oper. Res. 33 (2008), 821–838.