

## Lecture 9 - Part 1: Solving discounted stochastic games

Lecturer: Peter Bro Miltersen

Scribe: Marcel Keller

## 1 Algorithms for solving discounted stochastic games (continued)

Recall the strategy iteration algorithm from lecture 7.

---

**Algorithm 1** Strategy iteration
 

---

$x$  := arbitrary behaviour strategy for player I

**while** true **do**

$y$  := minimax strategy for player II in 1-player game where player I is forced to play  $x$

$\forall k : \alpha_k := u_1^k(x, y)$

$\forall k : x_k := \text{maximin}[A^{(k)}(\alpha_k)]$

**end while**

---

This algorithm can run forever, so we want to make sure that it at least approximates the solution. Then, we can return after a reasonable number of iterations. Let  $\alpha_k^{(t)}$  and  $x_k^{(t)}$  denote the value of  $\alpha_k$  and  $x_k$ , respectively, in the  $t$ -th iteration of the algorithm.

**Lemma 1**  $\alpha_k^{(t)}$  increases as  $t$  goes to infinity.

This was proven in lecture 8.

**Fact 2**  $\alpha_k^{(t)}$  converges to  $\text{val}(k)$  as  $t$  goes to infinity.

**Proof sketch** Since  $\alpha_k^{(t)}$  is increasing, it can either diverge to infinity or converge to  $v_k \in \mathbb{R}$ . The former is not possible because  $\alpha_k^{(t)}$  is always in the interval between the highest and lowest reward. It remains to show that  $v_k = \text{val}(k)$ .

Suppose that  $\alpha_k^{(t)}$  stabilizes, i.e.  $\alpha_k^{(t_0)} = \alpha_k^{(t_0+1)}$  for some  $t_0 \in \mathbb{N}$ , and thus,  $\alpha_k^{(t_0)}$  is a fix point of the iteration and  $\alpha_k^{(t)} = \alpha_k^{(t_0)}$  for all  $t \geq t_0$ . Hence,  $v_k = \alpha_k^{(t_0)}$ , and  $v_k = \text{val}(k)$  because  $(v_k)$  is also a fix point of value iteration. To see that, recall that player II cannot improve its outcome in the global game with player I sticking to  $(x_k)$ . Therefore, player II cannot improve in any of  $A^{(k)}(\alpha_k)$  with player I playing  $x_k$ , and thus,  $v_k$  is the value of  $A^{(k)}(\alpha_k)$  for all  $k$ .

For the case that  $\alpha_k^t$  converges without stabilizing, one can also prove that  $v_k = \lim_{t \rightarrow \infty} \alpha_k^{(t)} = \text{val}(k)$  using elementary analytical tools.  $\square$

At least heuristically, the convergence is faster than value iteration because  $y$  is chosen as a minimax strategy for the whole game (with player I sticking to  $(x_k)$ ) instead of just  $(A^{(k)}(\alpha_k))$ .

## 1.1 Important special case: Perfect information stochastic games

**Definition 3** *Perfect information stochastic games are stochastic games where each matrix is either a row or a column vector.*

In other words, only one player has an actual choice at any time in the game, there are no simultaneous moves, and every player has full information about the consequences of his decision. Therefore, a deterministic maximin strategies for the individual matrix games  $A^{(k)}(\alpha_k)$  consists of choosing the row with the best payoff. That is, we can let  $(x_k)$  be a positional (deterministic) strategy in strategy iteration.

As long as we have not stabilized, we will never see the same  $(x_k)$  again because  $(\alpha_k)$  increases. There are exactly  $T = \prod_{j=1}^N \cdot m_k$  positional strategies for player I, we thus stabilize in at most  $T$  iterations. Since every iteration involves solving  $N$  linear programs to compute  $(x_k)$ , the algorithm stops after a finite number of linear program computations.

**Theorem 4 (Oliver Friedman, LICS 2009)** *There is a family of games so that the number of iterations of strategy iteration is  $2^{\Omega(N)}$ .*

So unfortunately, strategy iteration is not a polynomial time algorithm, even for perfect information games. Before the above theorem was proven, strategy iteration for perfect information stochastic games was widely conjectured to be a polynomial time algorithm. This might be surprising because “nasty” exponential examples for the simplex algorithm (the Klee-Minty examples) were already found in the 70s. The following variant of the strategy iteration is an attempt to improve the running time, but it is not known whether it actually achieves an expected polynomial running time.

**Variant of strategy iteration** Replace the update of  $(x_k)$

$$\forall k : x_k := \text{maximin}[A^{(k)}(\alpha_k)]$$

by

$$\forall k \in L : x_k := \text{maximin}[A^{(k)}(\alpha_k)],$$

where  $L$  is a subset of  $\{1, \dots, N\}$ , randomly chosen in every iteration. The best known upper bound for unmodified strategy iteration, assuming  $m_k, n_k \in \{1, 2\} \forall k$ , is  $O(2^N/N)$  iterations. The best known upper bound for the expected number of iterations in the randomized version is  $O(2^{0.78N})$ , still assuming  $m_k, n_k \in \{1, 2\} \forall k$ . It is not quite clear at this point in time (at least not to the lecturer) if Friedman’s lower bound applies to the randomized variant of strategy improvement.

We do not know any polynomial time algorithm for solving perfect information stochastic games. The best known worst case expected time is achieved by RANDOMFACET. For the case of  $m_k, n_k \in \{1, 2\}$ , it has expected running time  $2^{O(\sqrt{N})}$ , i.e., a slightly subexponential running time. To be precise, the number of recursive calls is bounded by  $e^{2\sqrt{N}}$ .

---

**Algorithm 2** RANDOMFACET( $x, I$ )

---

**Require:**  $x$  is positional strategy

**Require:**  $I \subset \{1, \dots, N\}$

**Ensure:** Returns maximin strategy for game modified by  $x$  being frozen in  $I^c$ .

**if**  $I = \emptyset$  **then**

    return  $x$

**else**

    pick  $i \in I$  at random

$z := \text{RANDOMFACET}(x, I - \{i\})$

**if** One iteration of strategy iteration would not change  $z$  at  $i$  **then**

        return  $z$

**else**

        return  $\text{RANDOMFACET}(z^{(i)}, I - \{i\})$ , where  $z^{(i)}$  denotes  $z$  with the other choice in  $i$

**end if**

**end if**

---