Lecture notes for "Analysis of Algorithms": **Markov decision processes**

Lecturer: Thomas Dueholm Hansen [*]

June 26, 2013

Abstract

We give an introduction to *infinite-horizon Markov decision processes* (MDPs) with finite sets of *states* and *actions*. We focus primarily on *discounted* MDPs for which we present Shapley's (1953) *value iteration algorithm* and Howard's (1960) *policy iteration algorithm*. We also give a short introduction to discounted *turn-based stochastic games*, a 2-player generalization of MDPs. Finally, we give a short introduction to two alternative criteria for optimality: *average cost* and *total cost*.

The presentation given in these lecture notes is based on [6, 9, 5].

1 Markov decision processes

A Markov decision process (MDP) is composed of a finite set of *states*, and for each state a finite, non-empty set of *actions*. In each time unit, the MDP is in exactly one of the states. A *controller* must choose one of the actions associated with the current state. Using an action $a \in A$ incurs an immediate *cost*, and results in a probabilistic transition to a new state according to a probability distribution that depends on the action. The process goes on indefinitely. The goal of the controller is to *minimize* the incurred costs according to some criterion. We will later define three optimality criteria: *discounted cost*, *average cost*, and *total cost*.

Formally, a Markov decision process is defined as follows. We use $\Delta(S)$ to denote the set of probability distributions over elements of a set S.

Definition 1.1 (Markov decision process) *A* Markov decision process *(MDP) is a tuple* $M = (S, A, s, c, p)$*, where*

[*]School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E-mail: `tdh@cs.au.dk`.

- *S is a set of states,*
- *A is a set of actions,*
- *$s : A \to S$ assigns each action to the state from which it can be used,*
- *$c : A \to \mathbb{R}$ associates each action with a cost,*
- *and $p : A \to \Delta(S)$ associates each action with a probability distribution over states according to which the next state is chosen when a is used.*

For every $i \in S$, we let $A_i = \{a \in A \mid s(a) = i\}$ be the set of actions that can be used from i. We assume that $A_i \neq \emptyset$, for every $i \in S$. We use $n = |S|$ and $m = |A|$ to denote the number of states and actions, respectively.

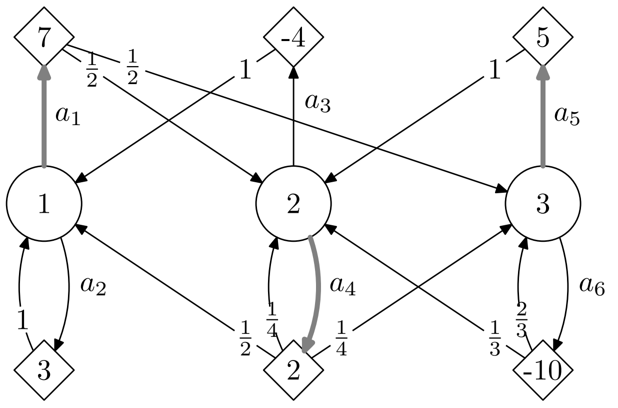
An MDP can be represented in different ways. A convenient and compact representation is the matrix representation given in Definition 1.2 below.

Definition 1.2 (Probability matrix, cost vector, and source matrix)
Let $M = (S, A, s, c, p)$ be an MDP. We may assume, without loss of generality, that $S = [n] = \{1, \ldots, n\}$ and $A = [m] = \{1, \ldots, m\}$.

- *We let $P \in \mathbb{R}^{m \times n}$, where $P_{a,i} = (p(a))_i$ is the probability of ending up in state i after taking action a, for every $a \in A = [m]$ and $i \in S = [n]$, be the* probability matrix *of M.*
- *We let $\mathbf{c} \in \mathbb{R}^m$, where $\mathbf{c}_a = c(a)$ is the cost of action $a \in A = [m]$, be the* cost vector *of M.*
- *We let $J \in \mathbb{R}^{m \times n}$, where $J_{a,i} = 1$ if $a \in A_i$ and $J_{a,i} = 0$ otherwise, be the* source matrix *of M.*

Figure 1 shows an example of an MDP and its matrix representation. The states are presented as circles numbered from 1 to 3. I.e., the set of states is $S = \{1, 2, 3\}$. The actions are represented as arrows leaving the states. The set of actions is $A = \{a_1, \ldots, a_6\}$. We, for instance, have $s(a_4) = 2$. The costs of the actions are shown inside the diamond-shaped vertices. For instance, $c(a_4) = 2$. Finally, the probability distribution associated with an action is shown as numbers labelling the edges leaving the corresponding diamond-shaped vertex. For instance, the probability of moving to state 1 when using action a_4 is $\frac{1}{2}$.

A *policy* for the controller of an MDP is a rule that specifies which action should be taken in each situation. The decision may in general depend on the current state of the process and possibly on the *history*, i.e., the sequence of states that have been visited and the actions that were used in those states. We also allow desicions to be randomized. We denote the set of histories by $\mathcal{H}$. The set of policies can be restricted in a natural way as follows. First, we



$$J = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \qquad P = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ 0 & 1 & 0 \\ 0 & \frac{1}{3} & \frac{2}{3} \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} 7 \\ 3 \\ -4 \\ 2 \\ 5 \\ -10 \end{bmatrix}$$

$$P_\pi = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ 0 & 1 & 0 \end{bmatrix} \qquad \mathbf{c}_\pi = \begin{bmatrix} 7 \\ 2 \\ 5 \end{bmatrix}$$

k	$\mathbf{e}_1^T P_\pi^k$		
0	1	0	0
1	0	$\frac{1}{2}$	$\frac{1}{2}$
2	$\frac{2}{8}$	$\frac{5}{8}$	$\frac{1}{8}$
3	$\frac{10}{32}$	$\frac{13}{32}$	$\frac{9}{32}$
$\vdots$		$\vdots$	

Figure 1: Example of a simple MDP and a (positional) policy π.

may require that every decision only depends on the current state and the time (the number of steps performed). We call such policies *time-dependent*. Second, we may require that every decision only depends on the current state. We call such policies *positional*. We also require that positional policies do not make use of randomization.

Definition 1.3 (Policies)

- *A* history-dependent policy π *is a mapping* $\pi : \mathcal{H} \to \Delta(A)$ *such that for every history* $h \in \mathcal{H}$*,* $\pi(h)$ *assigns positive probability only to actions that can be used from the final state of* h*. We denote the set of history-dependent policies by* $\Pi(\mathcal{H})$*.*

- *A* time-dependent policy π *is a mapping* $\pi : S \times \mathbb{N}_0 \to \Delta(A)$ *such that for every pair* $(i,t) \in S \times \mathbb{N}_0$*,* $\pi(i,t)$ *assigns positive probability only to actions that can be used from the state* i*. We denote the set of time-dependent policies by* $\Pi(\mathcal{T})$*.*

- *A* positional policy π *is a mapping* $\pi : S \to A$ *such that* $\pi(i) \in A_i$*, for every* $i \in S$*. We denote the set of positional policies by* $\Pi(\mathcal{P})$*.*

If an MDP starts in some state $i \in S$ and the controller uses a policy π, then the state reached after t steps is a random variable $X_\pi(i,t)$, and the action used from the t-th state is a random variable $Y_\pi(i,t)$.

The random variable $X_\pi(i,t)$ can be nicely described for positional policies. We identify a positional policy π with the set $\pi(S) \subseteq A$ of actions used in π. We let $P_\pi \in \mathbb{R}^{n \times n}$ be the matrix obtained by selecting the *rows* of P whose indices belong to π. We will assume that the actions are ordered according to states, such that for every policy π, by the above convention we have $J_\pi = I$; the identity matrix. Note that $(P_\pi)_i = P_{\pi(i)}$. Similarly, we let $\mathbf{c}_\pi \in \mathbb{R}^n$ be the vector containing the costs of the actions that belong to π. Note that P_π is a (row) *stochastic matrix*; its elements are non-negative and the elements in each row sum to 1. Thus, P_π defines a Markov chain, and we get that

$$\Pr\left[X_\pi(i,t) = j\right] \;=\; (P_\pi^t)_{i,j} \;.$$

In particular, if an MDP starts in some state i and the controller uses a positional policy π, then the probabilities of being in the different states after t steps are given by the vector $\mathbf{e}_i^T P_\pi^t$, where $\mathbf{e}_i$ is the i'th unit vector.

Figure 1 shows a positional policy π for a simple MDP. The policy is represented by bold gray arrows. The corresponding matrix P_π and vector $\mathbf{c}_\pi$ are also shown. Furthermore, the table at the lower right corner shows the probabilities of being in different states after k steps when starting in state 1.

Running an MDP with a (possibly history-dependent) policy π generates a random, infinite sequence of costs $c(Y_\pi(i,0)), c(Y_\pi(i,1)), c(Y_\pi(i,2)), \ldots$. This sequence of costs is evaluated according to some *optimality criterion* C, and the goal of the controller is to minimize the expected resulting value, $\mathrm{val}^C_\pi(i)$. We will use $\mathrm{val}^C(c_0, c_1, c_2, \dots)$ to denote the value of the infinite sequence $c_0, c_1, c_2, \dots$ according to the criterion C. We then have:

$$\mathrm{val}^C_\pi(i) \;=\; \mathrm{E}\left[\mathrm{val}^C(c(Y_\pi(i,0)), c(Y_\pi(i,1)), c(Y_\pi(i,2)), \dots)\right] \;.$$

Let us again note that things work out nicely for positional policies. In this case the pair $P_\pi, \mathbf{c}_\pi$ is a Markov chain with costs assigned to its states, and the expected cost observed at time t when starting from state i and using the positional policy π is then:

$$\mathrm{E}\left[c(Y_\pi(i,t))\right] \;=\; \mathbf{e}_i^T P_\pi^t \mathbf{c}_\pi \;. \tag{1}$$

Definition 1.4 (Discounted cost, average cost, and total cost) *Let* $0 < \gamma < 1$ *be a* discount factor. *The discounted cost criterion* $D(\gamma)$*, the average cost criterion* A*, and the total cost criterion* T *are defined by:*

$$\begin{aligned}
\mathrm{val}^{D(\gamma)}(c_0, c_1, c_2, \dots) &= \sum_{t=0}^{\infty} \gamma^t c_t \\
\mathrm{val}^{A}(c_0, c_1, c_2, \dots) &= \limsup_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N-1} c_t \\
\mathrm{val}^{T}(c_0, c_1, c_2, \dots) &= \sum_{t=0}^{\infty} c_t
\end{aligned}$$

Note that for the total cost criterion the series may diverge. We will later introduce a *stopping condition* that ensures convergence in this case.

Definition 1.5 (Optimal policy) *Let* $M = (S, A, s, c, p)$ *be an MDP and let* π *be a policy. We say that* π *is* optimal *for* M *with respect to a criterion* C *if and only if for all* $\pi' \in \Pi(\mathcal{H})$ *and all* $i \in S$ *we have* $\mathrm{val}^C_\pi(i) \leq \mathrm{val}^C_{\pi'}(i)$.

Note that an optimal policy must minimize the expected value for all states simultaneously. The existence of an optimal policy is non-trivial. We are, however, going to prove the following theorem that shows that, in fact, there always exists an optimal positional policy. The theorem was first proved by Shapley [12] in 1953 for discounted costs.

Theorem 1.6 *Every MDP has an optimal positional policy w.r.t. the discounted cost criterion, the average cost criterion, and the total cost criterion.*

Theorem 1.6 shows that when looking for an optimal policy we may restrict our search to positional policies. We will prove the theorem in the later sections. The following lemma makes the first step towards proving Theorem 1.6. It shows that we may restrict our search to time-dependent policies. More precisely, the lemma shows that for every history-dependent policy there exists a time-dependent policy that visits the same states and uses the same actions with the same probabilities.

Lemma 1.7 *Let $\pi \in \Pi(\mathcal{H})$ be any history-dependent policy, and let $i_0 \in S$ be any starting state. Then there exists a time-dependent policy $\pi' \in \Pi(\mathcal{T})$ such that for every non-negative integer $t \geq 0$, every state $i \in S$, and every action $a \in A$ we have $\Pr[X_{\pi'}(i_0, t) = i] = \Pr[X_\pi(i_0, t) = i]$ and $\Pr[Y_{\pi'}(i_0, t) = a] = \Pr[Y_\pi(i_0, t) = a]$.*

Proof: Since we are given π we can construct π' such for all $t \geq 0$, $i \in S$, and $a \in A$:

$$\Pr[\pi'(i, t) = a] \;=\; \Pr[Y_{\pi'}(i_0, t) = a \mid X_{\pi'}(i_0, t) = i] \;:=\; \Pr[Y_\pi(i_0, t) = a \mid X_\pi(i_0, t) = i] \;.$$

It follows that if $\Pr[X_{\pi'}(i_0, t) = i] = \Pr[X_\pi(i_0, t) = i]$, i.e., the two policies reach the same states in t steps with the same probabilities, then $\Pr[Y_{\pi'}(i_0, t) = a] = \Pr[Y_\pi(i_0, t) = a]$, i.e., the same actions are chosen from the t-th state with the same probabilities.

We next prove by induction in t that $\Pr[X_{\pi'}(i_0, t) = i] = \Pr[X_\pi(i_0, t) = i]$ for all $t \geq 0$ and $i \in S$. For $t = 0$ we have $\Pr[X_\pi(i_0, 0) = i_0] = \Pr[X_{\pi'}(i_0, 0) = i_0] = 1$ as desired. Recall that $P_{a,i}$ is the probability of moving to state i when using action a. For $t > 0$ we have by induction:

$$\begin{aligned}\Pr[X_{\pi'}(i_0, t) = i] \;&=\; \sum_{j \in S} \sum_{a \in A_j} \Pr[X_{\pi'}(i_0, t-1) = j] \cdot \Pr[\pi'(j, t-1) = a] \cdot P_{a,i} \\ &=\; \sum_{j \in S} \sum_{a \in A_j} \Pr[X_\pi(i_0, t-1) = j] \cdot \Pr[Y_\pi(i_0, t-1) = a \mid X_\pi(i_0, t-1) = j] \cdot P_{a,i} \\ &=\; \Pr[X_{\pi'}(i_0, t) = i] \;.\end{aligned}$$

□

2 The discounted cost criterion

Throughout this section we let $M = (S, A, s, c, p)$ be any given MDP, we let $(P, \mathbf{c}, J)$ be the matrix representation of M, and we let $0 < \gamma < 1$ be a given discount factor. There are two ways of interpreting the discount factor γ. The discount factor may be viewed as the inverse of the rate of inflation. Thus, a cost of $c(a)$ of action a at time t corresponds to a cost of $\gamma^t c(a)$ at time 0. Alternatively, $1-\gamma$ may be viewed as the probability that the MDP stops after each step. The expected cost of an action a at time t is then again $\gamma^t c(a)$, as γ^t is the probability that the MDP reaches time t.

Recall that for the discounted cost criterion we define $\mathrm{val}^{D(\gamma)}(c_0, c_1, c_2, \dots) = \sum_{t=0}^{\infty} \gamma^t c_t$, such that the expected total discounted cost when starting in a state i and using a policy π is

$$\mathrm{val}_\pi^{D(\gamma)}(i) \;=\; \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t c(Y_\pi(i, t))\right] \;=\; \sum_{t=0}^{\infty} \gamma^t \mathrm{E}\left[c(Y_\pi(i, t))\right] \;.$$

We leave it as an exercise to show that $\mathrm{val}_\pi^{D(\gamma)}(i)$ exists for every policy $\pi \in \Pi(\mathcal{H})$. In the important special case of positional policies we get, using (1):

$$\mathrm{val}_\pi^{D(\gamma)}(i) \;=\; \sum_{t=0}^{\infty} \mathbf{e}_i^T (\gamma P_\pi)^t \mathbf{c}_\pi \;.$$

The following lemma is helpful for understanding this infinite sum. We leave proving the lemma as an exercise.

Lemma 2.1 *For any policy π, the matrix $(I - \gamma P_\pi)$ is nonsingular and*

$$(I - \gamma P_\pi)^{-1} \;=\; \sum_{t=0}^{\infty} (\gamma P_\pi)^t \;\geq\; I \;.$$

Definition 2.2 (Value vector) *For every policy π, we define the* value vector $\mathbf{v}^\pi \in \mathbb{R}^n$ *by:*

$$\forall i \in S : \quad (\mathbf{v}^\pi)_i = \mathrm{val}_\pi^{D(\gamma)}(i) \;.$$

Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ be two vectors. We say that $\mathbf{u} \leq \mathbf{v}$ if and only if $\mathbf{u}_i \leq \mathbf{v}_i$, for every $1 \leq i \leq n$. We say that $\mathbf{u} < \mathbf{v}$ if and only if $\mathbf{u} \leq \mathbf{v}$ and $\mathbf{u} \neq \mathbf{v}$. We interpret $\mathbf{u} \geq \mathbf{v}$ and $\mathbf{u} > \mathbf{v}$ similarly. Hence, a policy π is optimal if and only if $\mathbf{v}^\pi \leq \mathbf{v}^{\pi'}$ for all $\pi' \in \Pi(\mathcal{H})$.

The following lemma follows from Lemma 2.1 and Definition 2.2.

Lemma 2.3 *For every positional policy π, we have:*

$$\mathbf{v}^\pi \;=\; (I - \gamma P_\pi)^{-1} \mathbf{c}_\pi \;.$$

2.1 The value iteration algorithm

We next introduce Shapley's value iteration algorithm [12] for solving discounted MDPs, and in the process of doing so we prove the existence of an optimal positional policy. To motivate the value iteration algorithm we first consider finite-horizon MDPs with discounted costs. A discounted, finite-horizon MDP terminates after a given finite number of steps, t, and the goal of the controller is to minimize the expected discounted sum of the incurred costs. Furthermore, for a given vector $\mathbf{v} \in \mathbb{R}^n$ we add an additional cost of $\gamma^t \mathbf{v}_i$ for terminating in state i.

It is not difficult to see that a finite-horizon MDP can be solved using dynamic programming. That is, after $t-1$ steps the best choice is from each state to use the action that locally minimizes the cost. This gives us a new vector v' corresponding to the costs of reaching the different states after $t-1$ steps. The process can then be repeated until all decisions are specified. The following definitions are useful for formalizing the process. Note that $P_a \in \mathbb{R}^{1 \times n}$ is the a-th row of the matrix P, and that $\mathbf{c}_a$ is the cost of action a.

Definition 2.4 (Value iteration operator) *Let $\mathbf{v} \in \mathbb{R}^n$ be an arbitrary vector. Define the* value iteration operator $\mathcal{T} : \mathbb{R}^n \to \mathbb{R}^n$ *by:*

$$\forall i \in S : \quad (\mathcal{T}\mathbf{v})_i \;=\; \min_{a \in A_i} \; \mathbf{c}_a + \gamma P_a \mathbf{v} \; .$$

Definition 2.5 (Policy extraction operator) *Define the* policy extraction operator $\mathcal{P}$: $\mathbb{R}^n \to \Pi(\mathcal{P})$ *to map any vector $\mathbf{v} \in \mathbb{R}^n$ to a positional policy $\pi = \mathcal{P}\mathbf{v}$ such that:*

$$\forall i \in S : \quad \pi(i) \;\in\; \operatorname*{argmin}_{a \in A_i} \; \mathbf{c}_a + \gamma P_a \mathbf{v} \; .$$

The following relation between the value iteration operator and the policy extraction operator is immediate.

Lemma 2.6 *For every $\mathbf{v} \in \mathbb{R}^n$ we have $\mathcal{T}\mathbf{v} = \mathbf{c}_\pi + \gamma P_\pi \mathbf{v}$, where $\pi = \mathcal{P}\mathbf{v}$.*

Following the above discussion, the minimum values that can be obtained using any (time-dependent) policy for a discounted, finite-horizon MDP that terminates after t steps with termination costs specified by $\mathbf{v} \in \mathbb{R}^n$ are $\mathcal{T}^t\mathbf{v}$. Furthermore, the actions used after k steps, for $k < t$, are those of the positional policy $\mathcal{P}\mathcal{T}^{t-k-1}\mathbf{v}$. The idea of the value iteration algorithm is to repeat the process for t going to infinity, or until t is sufficiently large.

For every time-dependent policy $\pi \in \Pi(\mathcal{T})$, and every integer $t \geq 0$. We let $\pi_{>t}$ be the policy obtained from π by cutting away the first t rounds, that is, $\pi_{>t}(i,k) = \pi(i,k+t)$ for all $i \in S$ and $k \geq 0$. Moreover, we let $\pi_{\Delta t}$ be the policy obtained from π by changing the decisions of the first t rounds such that for the k-th round, for $k < t$, we use the actions from the positional policy $\mathcal{P}\mathcal{T}^{t-k-1}\mathbf{v}^{\pi_{>t}}$. Note that the first t rounds of $\pi_{\Delta t}$ can viewed as an optimal policy for the finite-horizon MDP that terminates after t steps with termination costs $\mathbf{v}^{\pi_{>t}}$. In particular, the decisions made by $\pi_{\Delta t}$ during the first t rounds are at least as good as the corresponding decisions made by π. We get the following lemma.

Lemma 2.7 *Let $\pi \in \Pi(\mathcal{T})$ be any time-dependent policy, and let $t \geq 0$ be an integer. Then $\mathbf{v}^{\pi_{\Delta t}} \;=\; \mathcal{T}^t\mathbf{v}^{\pi_{>t}} \;\leq\; \mathbf{v}^\pi$.*

We next show that $\mathbf{v}^{\pi_{\Delta t}}$ always converges to unique vector $\mathbf{v}^*$, regardless of the policy π, and that $\mathcal{P}\mathbf{v}^*$ is an optimal positional policy. First we show that the operator $\mathcal{T}$ is a contraction with Lipschitz constant γ.

Lemma 2.8 *For every $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ we have:*

$$\|\mathcal{T}\mathbf{u} - \mathcal{T}\mathbf{v}\|_\infty \;\leq\; \gamma \, \|\mathbf{u} - \mathbf{v}\|_\infty \; .$$

Proof: Let $i \in S$, and assume that $(\mathcal{T}\mathbf{u})_i \geq (\mathcal{T}\mathbf{v})_i$. Let

$$a \in \operatorname*{argmin}_{a \in A_i} \mathbf{c}_a + \gamma P_a \mathbf{u} \qquad \text{and} \qquad b \in \operatorname*{argmin}_{b \in A_i} \mathbf{c}_b + \gamma P_b \mathbf{v} \;.$$

Then,

$$\begin{aligned} (\mathcal{T}\mathbf{u} - \mathcal{T}\mathbf{v})_i &= (\mathbf{c}_a + \gamma P_a \mathbf{u}) - (\mathbf{c}_b + \gamma P_b \mathbf{v}) \\ &\leq (\mathbf{c}_b + \gamma P_b \mathbf{u}) - (\mathbf{c}_b + \gamma P_b \mathbf{v}) \\ &= \gamma P_b (\mathbf{u} - \mathbf{v}) \\ &\leq \gamma \left\| \mathbf{u} - \mathbf{v} \right\|_\infty \;. \end{aligned}$$

The last inequality follows from the fact that the elements in P_b are non-negative and sum up to 1. The case when $(\mathcal{T}\mathbf{u})_i \leq (\mathcal{T}\mathbf{v})_i$ is analogous.

Since the inequality holds for all states i, it holds for the state for which the absolute difference of $\mathcal{T}\mathbf{u}$ and $\mathcal{T}\mathbf{v}$ is largest, and, hence, the result follows. $\square$

The Banach fixed point theorem now implies that the operator $\mathcal{T}$ has a unique fixed point.

Corollary 2.9 *There is a unique vector $\mathbf{v}^* \in \mathbb{R}^n$ such that $\mathcal{T}\mathbf{v}^* = \mathbf{v}^*$.*

The following lemma shows that the policy $\pi^* = \mathcal{P}\mathbf{v}^*$ is optimal, which proves Theorem 1.6 for the discounted cost criterion. Note that all optimal policies must, per definition, have the same value vectors. We therefore refer to $\mathbf{v}^*$ as the *optimal value vector*.

Lemma 2.10 *Let $\mathbf{v}^* \in \mathbb{R}^n$ be the unique fixed point of $\mathcal{T}$, and let $\pi^* = \mathcal{P}\mathbf{v}^*$. Then, π^* is an optimal policy, and $\mathbf{v}^{\pi^*} = \mathbf{v}^*$.*

Proof: We first show that $\mathbf{v}^{\pi^*} = \mathbf{v}^*$. Since $\mathbf{v}^*$ is a fixed point for $\mathcal{T}$ and $\pi^* = \mathcal{P}\mathbf{v}^*$ we get from Lemma 2.6 that:

$$\mathbf{v}^* \;=\; \mathcal{T}\mathbf{v}^* \;=\; \mathbf{c}_{\pi^*} + \gamma P_{\pi^*} \mathbf{v}^*$$

We know from Lemma 2.1 that the matrix $(I - \gamma P_{\pi^*})$ is non-singular, which implies that $\mathbf{v}^* = (I - \gamma P_{\pi^*})^{-1} \mathbf{c}_{\pi^*}$. Lemma 2.3 then shows that $\mathbf{v}^{\pi^*} = \mathbf{v}^*$.

We next show that π^* is an optimal policy. We know from Lemma 2.7 that for every time-dependent policy $\pi \in \Pi(\mathcal{T})$ and every integer $t \geq 0$, we have $\mathbf{v}^{\pi_{\Delta t}} = \mathcal{T}^t \mathbf{v}^{\pi_{>t}} \leq \mathbf{v}^\pi$. We will show that $\mathcal{T}^t \mathbf{v}^{\pi_{>t}} \to \mathbf{v}^*$ for $t \to \infty$. It then follows that $\mathbf{v}^* \leq \mathbf{v}^\pi$, which shows that π^* is optimal.

Let $\mathbf{c}_{\max} = \max_{a \in A} |\mathbf{c}_a|$ be largest absolute value of the cost of any action. Observe first that for every policy π and every state $i \in S$, we have $|(\mathbf{v}^\pi)_i| \leq \frac{\mathbf{c}_{\max}}{1-\gamma}$. Indeed, $\mathbf{c}_{\max}$ is the largest absolute value of any cost incurred by a single step of the MDP, and if the cost is $\mathbf{c}_{\max}$ at every step the total discounted cost is $\sum_{t=0}^{\infty} \gamma^t \mathbf{c}_{\max} = \frac{\mathbf{c}_{\max}}{1-\gamma}$. From repeated use of Lemma 2.8 we then get:

$$\left\| \mathcal{T}^t \mathbf{v}^{\pi_{>t}} - \mathbf{v}^* \right\|_\infty \;=\; \left\| \mathcal{T}^t \mathbf{v}^{\pi_{>t}} - \mathcal{T}^t \mathbf{v}^* \right\|_\infty \;\leq\; \gamma^t \left\| \mathbf{v}^{\pi_{>t}} - \mathbf{v}^* \right\|_\infty \;\leq\; \gamma^t \frac{2\mathbf{c}_{\max}}{1-\gamma} \;,$$

```
Function ValueIteration($\mathbf{u}, \epsilon$)
    while $\|\mathbf{u} - \mathcal{T}\mathbf{u}\|_\infty > \frac{\epsilon}{2}(1-\gamma)$ do
        $\mathbf{u} \leftarrow \mathcal{T}\mathbf{u}$;
    return $\mathbf{u}$;
```

Figure 2: The value iteration algorithm.

and, hence, $\mathcal{T}^t \mathbf{v}^{\pi_{>t}} \to \mathbf{v}^*$ for $t \to \infty$.

□

The ValueIteration algorithm, given in Figure 2, repeatedly applies the value iteration operator $\mathcal{T}$ to an initial vector $\mathbf{u} \in \mathbb{R}^n$ until the difference between two successive vectors is sufficiently small, i.e., $\|\mathbf{u} - \mathcal{T}\mathbf{u}\|_\infty \leq \frac{\epsilon}{2}(1-\gamma)$ for some given $\epsilon \geq 0$.

The value iteration algorithm was introduced for MDPs by Bellman [1] in 1957. It can be viewed as a special case of an algorithm by Shapley [12] from 1953 for solving the more general class of *stochastic games*. Note that the vector $\mathcal{T}^t\mathbf{u}$ can be viewed as the optimal value vector for the finite-horizon MDP that terminates after t steps with termination costs specified by $\mathbf{u}$. As noted in the beginning of the section, the value iteration algorithm can therefore be viewed as a dynamic programming algorithm.

We say that a policy π is ϵ-optimal if $\|\mathbf{v}^\pi - \mathbf{v}^*\|_\infty \leq \epsilon$, for some $\epsilon \geq 0$. The following lemma shows that when the value iteration algorithm terminates we can extract an ϵ-optimal policy from the resulting vector.

Lemma 2.11 *Let $\mathbf{u} \in \mathbb{R}^n$ be a vector, and let $\pi = \mathcal{P}\mathbf{u}$ be the policy extracted from $\mathbf{u}$. If $\|\mathbf{u} - \mathcal{T}\mathbf{u}\|_\infty \leq \frac{\epsilon}{2}(1-\gamma)$, then $\|\mathbf{u} - \mathbf{v}^*\|_\infty \leq \frac{\epsilon}{2}$ and $\|\mathbf{u} - \mathbf{v}^\pi\|_\infty \leq \frac{\epsilon}{2}$. In particular, π is ϵ-optimal.*

Proof: We only prove that if $\|\mathbf{u} - \mathcal{T}\mathbf{u}\|_\infty \leq \frac{\epsilon}{2}(1-\gamma)$ then $\|\mathbf{u} - \mathbf{v}^*\|_\infty \leq \frac{\epsilon}{2}$. The second part of the lemma can be proved analogously. Indeed, if we consider the MDP where only the actions of π are available, then $\mathbf{v}^\pi = \mathbf{v}^*$.

We know from Lemma 2.8 that $\|\mathcal{T}\mathbf{u} - \mathbf{v}^*\|_\infty \leq \gamma\|\mathbf{u} - \mathbf{v}^*\|_\infty$. From the triangle inequality we get:

$$\|\mathbf{u} - \mathbf{v}^*\|_\infty \ \leq \ \|\mathbf{u} - \mathcal{T}\mathbf{u}\|_\infty + \|\mathcal{T}\mathbf{u} - \mathbf{v}^*\|_\infty \ \leq \ \frac{\epsilon}{2}(1-\gamma) + \gamma\|\mathbf{u} - \mathbf{v}^*\|_\infty \quad \Rightarrow$$

$$\|\mathbf{u} - \mathbf{v}^*\|_\infty \ \leq \ \frac{\epsilon}{2}\,.$$

□

Lemma 2.12 *For a given vector* $\mathbf{u} \in \mathbb{R}^n$*, let* $N \geq \frac{1}{1-\gamma} \log \frac{4\|\mathbf{u}-\mathbf{v}^*\|_\infty}{\epsilon(1-\gamma)}$ *be an integer. Then* $\|\mathcal{T}^N\mathbf{u} - \mathcal{T}^{N+1}\mathbf{u}\|_\infty \leq \frac{\epsilon}{2}(1-\gamma)$

Proof: We will show that $\|\mathcal{T}^N\mathbf{u}-\mathbf{v}^*\|_\infty \leq \frac{\epsilon}{4}(1-\gamma)$. Since, by Lemma 2.8, $\|\mathcal{T}^{N+1}\mathbf{u}-\mathbf{v}^*\|_\infty \leq \gamma\|\mathcal{T}^N\mathbf{u} - \mathbf{v}^*\|_\infty$, it then follows from the triangle inequality that:

$$\|\mathcal{T}^N\mathbf{u} - \mathcal{T}^{N+1}\mathbf{u}\|_\infty \;\leq\; \|\mathcal{T}^N\mathbf{u} - \mathbf{v}^*\|_\infty + \|\mathcal{T}^{N+1}\mathbf{u} - \mathbf{v}^*\|_\infty \;\leq\; \frac{\epsilon}{2}(1-\gamma)\;.$$

From repeated use of Lemma 2.8 we know that $\|\mathcal{T}^N\mathbf{u} - \mathbf{v}^*\|_\infty \leq \gamma^N\|\mathbf{u} - \mathbf{v}^*\|_\infty$. From the inequality $\log_{1/\gamma} x = \frac{\log x}{\log 1/\gamma} \leq \frac{\log x}{1-\gamma}$, we get that $N \geq \log_{1/\gamma} \frac{4\|\mathbf{u}-\mathbf{v}^*\|_\infty}{\epsilon(1-\gamma)}$. It follows that $\gamma^N\|\mathbf{u} - \mathbf{v}^*\|_\infty \leq \frac{\epsilon}{4}(1-\gamma)$. □

Lemmas 2.11 and 2.12 show that ValueIteration$(\mathbf{u}, \epsilon)$ computes a vector from which we can extract an ϵ-optimal policy. Furthermore, Lemma 2.12 shows that the number of iterations needed to find this vector is at most $O(\frac{1}{1-\gamma} \log \frac{\|\mathbf{u}-\mathbf{v}^*\|_\infty}{\epsilon(1-\gamma)})$. We next estimate the ϵ for which the resulting policy is guaranteed to be optimal.

For every MDP M, let $L(M, \gamma)$ be the number of bits needed to describe the matrix $(J-\gamma P)$ and the vector $\mathbf{c}$. More precisely, for every number $a = p/q$, where $p \in \mathbb{Z}$ and $q \in \mathbb{N}$ are relatively prime, we use $1 + \lceil\log_2(|p|+1)\rceil + \lceil\log_2(q+1)\rceil$ bits. Using Cramer's rule and simple bounds on the size of determinants, one can prove that the number of bits needed to describe a component of the value vector $\mathbf{v}^\pi = (I - \gamma P_\pi)^{-1}\mathbf{c}_\pi$, for some policy π, is at most $4L(M, \gamma)$. We leave this as an exercise.

Lemma 2.13 *Let* π *and* π' *be two policies such that* $\mathbf{v}^\pi \neq \mathbf{v}^{\pi'}$*. Then* $\|\mathbf{v}^\pi - \mathbf{v}^{\pi'}\|_\infty \geq 2^{-4L(M,\gamma)}$*.*

Corollary 2.14 *If* π *is* ϵ*-optimal for* $0 \leq \epsilon < 2^{-4L(M,\gamma)-1}$*, then* π *is optimal.*

Let $\mathbf{c}_{\max} = \max_{a\in A} |\mathbf{c}_a|$. As we saw in the proof of Lemma 2.10, it is not difficult to show that for every policy π and every state i, we have $|(\mathbf{v}^\pi)_i| \leq \frac{\mathbf{c}_{\max}}{1-\gamma}$. We also clearly have $\mathbf{c}_{\max} \leq 2^{L(M,\gamma)}$. Hence, $\|\mathbf{v}^* - \mathbf{0}\|_\infty \leq (2^{L(M,\gamma)})/(1-\gamma)$, where $\mathbf{0}$ is the all-zero vector. Combining Lemma 2.12 and Corollary 2.14 we then get the following bound on the number of iterations needed for the value iteration algorithm to produce an optimal policy:

Theorem 2.15 *Let* $\pi = \mathcal{P}\mathcal{T}^N\mathbf{0}$*, where* $N = \frac{1}{1-\gamma} \log \frac{8 \cdot 2^{5L(M,\gamma)}}{(1-\gamma)^2} \leq O(\frac{L(M,\gamma)}{1-\gamma} \log \frac{1}{1-\gamma})$*. Then* π *is optimal.*

2.2 The policy iteration algorithm

We next give an introduction to Howard's policy iteration algorithm [7] from 1960. The policy iteration algorithm can be viewed as an attempt to speed up the value iteration algorithm. It is the most widely used algorithm for solving MDPs in practice. As we will see in Section 2.3 below it can be viewed as a generalization of the simplex method for solving linear programs in the context of MDPs.

Throughout this section we restrict our attention to positional policies. Hence, whenever we use the term policy we mean a positional policy.

Definition 2.16 (One-step operator) *Let π be a policy and $\mathbf{v} \in \mathbb{R}^n$ be a vector. We define the* one-step operator $\mathcal{T}_\pi : \mathbb{R}^n \to \mathbb{R}^n$ *by:*

$$\mathcal{T}_\pi \mathbf{v} \;=\; \mathbf{c}_\pi + \gamma P_\pi \mathbf{v}\ .$$

Recall that Lemma 2.3 says that $\mathbf{v}^\pi = (I - \gamma P_\pi)^{-1}\mathbf{c}_\pi$. Hence, for every policy π, the value vector $\mathbf{v}^\pi$ is a fixed point for $\mathcal{T}_\pi$. The following lemma can be proved in the same way as Lemma 2.8 and Corollary 2.9.

Lemma 2.17 *For every policy π, $\mathbf{v}^\pi$ is a unique fixed point for $\mathcal{T}_\pi$, and for every vector $\mathbf{u} \in \mathbb{R}^n$, $\mathcal{T}_\pi^t \mathbf{u} \to \mathbf{v}^\pi$ for $t \to \infty$.*

Lemma 2.18 *Let π and π' be two policies. If $\mathcal{T}_{\pi'}\mathbf{v}^\pi < \mathbf{v}^\pi$, then $\mathbf{v}^{\pi'} \le \mathcal{T}_{\pi'}\mathbf{v}^\pi < \mathbf{v}^\pi$. The same holds for $\le$, $>$, and $\ge$.*

Proof: We only prove the lemma for $<$. The other cases are proved analogously.

First observe that for any vector $\mathbf{u} \in \mathbb{R}^n$, if $\mathcal{T}_{\pi'}\mathbf{u} \le \mathbf{u}$ then since all entries of $P_{\pi'}$ are non-negative we get:

$$\mathcal{T}_{\pi'}^2 \mathbf{u} \;=\; \mathbf{c}_{\pi'} + \gamma P_{\pi'}(\mathcal{T}_{\pi'}\mathbf{u}) \;\le\; \mathbf{c}_{\pi'} + \gamma P_{\pi'}\mathbf{u} \;=\; \mathcal{T}_{\pi'}\mathbf{u}\ .$$

If $\mathcal{T}_{\pi'}\mathbf{v}^\pi \le \mathbf{v}^\pi$, then repeated use of the above inequality shows that $\mathbf{v}^\pi \ge \mathcal{T}_{\pi'}\mathbf{v}^\pi \ge \mathcal{T}_{\pi'}^2\mathbf{v}^\pi \ge \mathcal{T}_{\pi'}^3\mathbf{v}^\pi \ge \dots$. Since, by Lemma 2.17, $\mathcal{T}_{\pi'}^t\mathbf{v}^\pi \to \mathbf{v}^{\pi'}$ for $t \to \infty$, it follows that $\mathbf{v}^\pi \ge \mathcal{T}_{\pi'}\mathbf{v}^\pi \ge \mathbf{v}^{\pi'}$. In particular, if $\mathcal{T}_{\pi'}\mathbf{v}^\pi < \mathbf{v}^\pi$ then $\mathbf{v}^{\pi'} < \mathbf{v}^\pi$. $\square$

Lemma 2.19 *Let π be a policy. If for every policy π' we have $\mathcal{T}_{\pi'}\mathbf{v}^\pi \not< \mathbf{v}^\pi$ then π is optimal.*

Proof: We show that if for every policy π' we have $\mathcal{T}_{\pi'}\mathbf{v}^\pi \not< \mathbf{v}^\pi$, then for every policy π' we have $\mathcal{T}_{\pi'}\mathbf{v}^\pi \ge \mathbf{v}^\pi$. It then follows from Lemma 2.18 that $\mathbf{v}^{\pi'} \ge \mathbf{v}^\pi$ for all policies π'.

Assume for the sake of contradiction that there exists a policy π' such that $(\mathcal{T}_{\pi'}\mathbf{v}^\pi)_i < (\mathbf{v}^\pi)_i$ for some state i. We will construct a policy π'' such that $\mathcal{T}_{\pi''}\mathbf{v}^\pi < \mathbf{v}^\pi$ which gives a contradiction. Let π'' be defined by $\pi''(i) = \pi'(i)$ and $\pi''(j) = \pi(j)$ for all $j \ne i$. Then, $(\mathcal{T}_{\pi''}\mathbf{v}^\pi)_i < (\mathbf{v}^\pi)_i$ and $(\mathcal{T}_{\pi''}\mathbf{v}^\pi)_j = (\mathbf{v}^\pi)_j$ for all $j \ne i$. $\square$

Note that $\mathcal{T}_{\pi'}\mathbf{v}^\pi < \mathbf{v}^\pi$ can be equivalently stated as $\mathbf{c}_{\pi'} - (I - \gamma P_{\pi'})\mathbf{v}^\pi < \mathbf{0}$. I.e., if an action is better for one step with respect to the values of the current policy, then it is also better to keep using this action. This motivates the following definition.

```
Function PolicyIteration($\pi$)
    while $\exists \pi' : \mathcal{T}_{\pi'} \mathbf{v}^\pi < \mathbf{v}^\pi$ do
        $\pi \leftarrow \pi'$;
    return $\pi$;
```

```
Function PolicyIteration($\pi$)
    while $\exists$ an improving switch w.r.t. $\pi$ do
        Update $\pi$ by performing improving switches;
    return $\pi$;
```

```
Function Howard's PolicyIteration($\pi$)
    while $\mathcal{T}\mathbf{v}^\pi < \mathbf{v}^\pi$ do
        $\pi \leftarrow \mathcal{P}\mathbf{v}^\pi$;
    return $\pi$;
```

Figure 3: Two equivalent formulations of the policy iteration algorithm (top and middle), and the policy iteration algorithm with Howard's improvement rule (bottom).

Definition 2.20 (Reduced costs, improving switches) *The* reduced cost *vector* $\bar{\mathbf{c}}^\pi \in \mathbb{R}^m$ *corresponding to a policy* π *is defined to be*

$$\bar{\mathbf{c}}^\pi \; = \; \mathbf{c} - (J - \gamma P)\mathbf{v}^\pi \; .$$

We say that an action $a \in A$ *is an* improving switch *with respect to a policy* π *if and only if* $\bar{\mathbf{c}}^\pi_a < 0$.

Note that actions $a \in \pi$ have reduced cost $\bar{\mathbf{c}}^\pi_a = 0$. We say that a policy π' is obtained from π by *performing improving switches* if every new action $a \in \pi' \setminus \pi$ is an improving switch with respect to π, i.e., if $(\bar{\mathbf{c}}^\pi)_a < 0$. We will use the notation $\pi' = \pi[B]$ where $B = \pi' \setminus \pi$. Lemma 2.18 can then be interpreted as saying that if a policy π' is obtained from π by performing improving switches then $\mathbf{v}^{\pi'} < \mathbf{v}^\pi$. On the other hand, Lemma 2.19 says that if there are no improving switches with respect to a policy π, then π is optimal. Hence, a policy is optimal if and only if there are no improving switches.

The PolicyIteration algorithm is given in Figure 3. It starts with some initial policy π^0 and generates an improving sequence $\pi^0, \pi^1, \ldots, \pi^N$ of policies, ending with an optimal policy π^N. In each iteration the algorithm *evaluates* the current policy π^k and computes the value vector $\mathbf{v}^{\pi^k}$ by solving a system of linear equations. The next policy π^{k+1} is obtained from π^k by performing a non-empty set of improving switches $B \subseteq A$ with respect to π^k, such that $\pi^{k+1} = \pi^k[B]$.

It follows from Lemma 2.18 that the value vectors strictly improve with each iteration; $\mathbf{v}^{k+1} < \mathbf{v}^k$. Hence, the number of iterations is bounded by the number of policies. Moreover, since there are no improving switches with respect to the final policy π^N, we know from Lemma 2.19 that π^N is optimal. We get:

Theorem 2.21 *For every initial policy π,* PolicyIteration(π) *terminates after a finite number of iterations, returning an optimal policy.*

The set of improving switches that is performed is decided by an *improvement rule*. There is, in fact, a whole family of policy iteration algorithms using different improvement rules. The most natural variant is, perhaps, the one in which the algorithm selects the improving switch with most negative reduced cost from every state and performs all these switches simultaneously, i.e., $\pi^{k+1} = \mathcal{P}\mathbf{v}^{\pi^k}$. This was the original improvement rule suggested by Howard [7], and we will refer to the algorithm obtained by using this improvement rule as Howard's policy iteration algorithm.

For the remainder of this section we focus on Howard's policy iteration algorithm. Let π^0 be some initial policy. We next relate the sequences of value vectors obtained by running Howard's policy iteration algorithm and the value iteration algorithm. The following lemma appears, e.g., in Meister and Holzbaur [8]. We use $\mathbf{v}^k$ as short-hand notation for $\mathbf{v}^{\pi^k}$.

Lemma 2.22 *Let π^0 be any policy, and let $(\mathbf{v}^k)_{k=0}^N$ be the value vectors generated by Howard's policy iteration algorithm starting from π^0. Then, $\mathbf{v}^k \leq \mathcal{T}^k\mathbf{v}^0$, for every $0 \leq k \leq N$.*

Proof: We prove the lemma by induction. Clearly the statement is true for $k = 0$. Suppose now that $\mathbf{v}^k \leq \mathcal{T}^k\mathbf{v}^0$. Since π^{k+1} is obtained from π^k by performing improving switches we know that $\mathcal{T}_{\pi^{k+1}}\mathbf{v}^{\pi^k} < \mathbf{v}^{\pi^k}$. Furthermore, since $\pi^{k+1} = \mathcal{P}\mathbf{v}^k$, we have $\mathcal{T}\mathbf{v}^k = \mathcal{T}_{\pi^{k+1}}\mathbf{v}^k$. Using Lemma 2.18, it then follows that $\mathbf{v}^{k+1} \leq \mathcal{T}_{\pi^{k+1}}\mathbf{v}^{\pi^k} = \mathcal{T}\mathbf{v}^k$. Finally, from the induction hypothesis and the monotonicity of the value iteration operator we get $\mathbf{v}^{k+1} \leq \mathcal{T}^{k+1}\mathbf{v}^0$. $\square$

Lemma 2.22 shows that Howard's policy iteration algorithm converges to the optimal value vector at least as fast that the value iteration algorithm.

Combining Lemma 2.22 with Theorem 2.15 we get the following theorem. In Theorem 2.15 the value iteration algorithm is initialized with the all-zero vector. It is not difficult to see, however, that the value vector of any policy $\mathbf{v}^{\pi^0}$ works as well. I.e., $\|\mathbf{v}^{\pi^0} - \mathbf{v}^*\|_\infty \leq (2^{L(M,\gamma)+1})/(1-\gamma)$, for any policy π^0.

Theorem 2.23 *Starting with any policy π, the number of iterations performed by Howard's policy iteration algorithm is at most $O(\frac{L(M,\gamma)}{1-\gamma}\log\frac{1}{1-\gamma})$.*

A recent series of papers by Ye [13]; Hansen, Miltersen, and Zwick [6]; and Scherrer [11] has shown the following improved bound. Note that the bound is *strongly polynomial* when γ is a fixed constant, i.e., the bound does not depend on the bit complexity $L(M,\gamma)$. In fact, the bound is linear in m, the number of actions, when γ is fixed.

Theorem 2.24 *Starting with any policy π, the number of iterations performed by Howard's policy iteration algorithm is at most $O(\frac{m}{1-\gamma}\log\frac{1}{1-\gamma})$.*

2.3 Linear programming formulation

The proofs given in this section were not covered during the lectures.

In this section we show how the problem of solving a discounted MDP can be formulated as a linear program. The linear program we present is due to d'Epenoux [2] from 1963.

We will no longer restrict our attention to positional policies. Recall that $Y_\pi(i,t)$ is the random variable corresponding to the action used after t steps when starting from state i and using policy π. We start by making the following useful definition of the *flux vector* of a policy π. The flux vector counts the expected number of times every action is used when we sum over all starting states, and where we interpret $1-\gamma$ as a stopping probability.

Definition 2.25 (Flux vector) *For every policy π, define the* flux vector $\mathbf{x}^\pi \in \mathbb{R}^m$ *by:*

$$\forall a \in A : \quad (\mathbf{x}^\pi)_a \;=\; \sum_{i\in S}\sum_{t=0}^{\infty} \gamma^t \Pr\left[Y_\pi(i,t)=a\right] \ .$$

We let $\mathbf{e}=(1,1,\ldots,1)^T \in \mathbb{R}^n$ be an all one vector. When π is a positional policy we use $\bar{\mathbf{x}}^\pi = (\mathbf{x}^\pi)_\pi \in \mathbb{R}^n$ to denote the vector obtained from $\mathbf{x}^\pi$ by selecting the entries corresponding to actions used in π. Note that for positional policies we have $(\mathbf{x}^\pi)_a = 0$ for $a \notin \pi$. Also note that for every positional policy π, every $i,j\in S$, and every $t \ge 0$, we have $\Pr\left[Y_\pi(i,t)=\pi(j)\right] = \Pr\left[X_\pi(i,t)=j\right] = (P_\pi^t)_{i,j}$. Hence,

$$\forall j \in S : \quad (\mathbf{x}^\pi)_{\pi(j)} \;=\; (\bar{\mathbf{x}}^\pi)_j \;=\; \sum_{i\in S}\sum_{t=0}^{\infty}\gamma^t\,(P_\pi^t)_{i,j} \;=\; \mathbf{e}^T\sum_{t=0}^{\infty}(\gamma P_\pi)^t\mathbf{e}_j \ ,$$

and from Lemma 2.1 we get:

Lemma 2.26 *For every positional policy π we have:*

$$(\bar{\mathbf{x}}^\pi)^T \;=\; \mathbf{e}^T(I-\gamma P_\pi)^{-1}$$

Note that since an optimal policy simultaneously minimizes the values of all states, it also minimizes the sum of the values of the states. Flux vectors provide an alternative way of summing the values of all the states:

Lemma 2.27 *For every policy π, we have*

$$\mathbf{e}^T\mathbf{v}^\pi = \mathbf{c}^T\mathbf{x}^\pi \ .$$

Proof: Recall that:

$$(\mathbf{v}^\pi)_i \;=\; \sum_{t=0}^{\infty} \gamma^t \,\mathrm{E}\left[c(Y_\pi(i,t))\right] \;=\; \sum_{t=0}^{\infty} \gamma^t \sum_{a\in A} \mathbf{c}_a \Pr\left[Y_\pi(i,t)=a\right] \;.$$

Hence we have:

$$\begin{aligned}
\mathbf{e}^T\mathbf{v}^\pi \;&=\; \sum_{i\in S}\sum_{t=0}^{\infty}\gamma^t \sum_{a\in A}\mathbf{c}_a \Pr\left[Y_\pi(i,t)=a\right] \\
&=\; \sum_{a\in A}\mathbf{c}_a \sum_{i\in S}\sum_{t=0}^{\infty}\gamma^t \Pr\left[Y_\pi(i,t)=a\right] \\
&=\; \sum_{a\in A}\mathbf{c}_a\,(\mathbf{x}^\pi)_a \;=\; \mathbf{c}^T\mathbf{x}^\pi \;.
\end{aligned}$$

□

Note that for every policy π and every state i, the expected number of times we use actions leaving i must be equal to the expected number of times we reach i. Also, no action can be used a negative number of times. Hence, the flux vector must satisfy $\mathbf{x}^\pi \geq \mathbf{0}$ and

$$\forall i \in S: \quad \sum_{a\in A_i}(\mathbf{x}^\pi)_a \;=\; 1 + \sum_{a\in A}(\mathbf{x}^\pi)_a\,\gamma P_{a,i} \;. \tag{2}$$

Using matrix notation, this equality can be stated as $(J-\gamma P)^T\mathbf{x}^\pi = \mathbf{e}$. This leads us to the following definition of a linear program (P) and its dual (D). The variables of the primal linear program (P) correspond to flux vectors, and the variables of the dual linear program (D) correspond to value vectors.

$$(P)\;\begin{array}{lrcl} \min & & \mathbf{c}^T\mathbf{x} & \\ s.t. & (J-\gamma P)^T\mathbf{x} & = & \mathbf{e} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array} \qquad (D)\;\begin{array}{lrcl} \max & & \mathbf{e}^T\mathbf{y} & \\ s.t. & (J-\gamma P)\mathbf{y} & \leq & \mathbf{c} \end{array}$$

The above discussion proves the following lemma.

Lemma 2.28 *For every policy π, the flux vector $\mathbf{x}^\pi$ is a feasible solution to the linear program (P).*

Lemma 2.29 *For every feasible solution $\mathbf{x}$ to the linear program (P) there exists a policy π with flux vector $\mathbf{x}^\pi = \mathbf{x}$.*

Proof: We construct π from $\mathbf{x}$ as follows. We let π be a time-dependent policy that uses the same decisions in every round. Define $\bar{\mathbf{x}} \in \mathbb{R}^n$ by $\bar{\mathbf{x}}_i = \sum_{a\in A_i}\mathbf{x}_a$ for all $i \in S$. We define π by:

$$\forall t \geq 0\;\forall i \in S\;\forall a \in A_i: \quad \Pr\left[\pi(i,t)=a\right] \;=\; \frac{\mathbf{x}_a}{\bar{\mathbf{x}}_i} \;.$$

Note that since the right-hand-side of (2) is at least 1, $\bar{\mathbf{x}}_i \geq 1$ for all i, and π is well-defined. Observe also that since π uses the same (randomized) decisions in every round it defines a Markov chain given by the stochastic matrix $Q \in \mathbb{R}^{n\times n}$ where $Q_j = \frac{1}{\bar{\mathbf{x}}_j}\sum_{a\in A_j} \mathbf{x}_a P_a$, for all states j. Define $\bar{\mathbf{x}}^\pi \in \mathbb{R}^n$ by $(\bar{\mathbf{x}}^\pi)_i = \sum_{a\in A_i}(\mathbf{x}^\pi)_a$ for all i, i.e., $(\bar{\mathbf{x}}^\pi)_i$ is the expected number of times we visit state i. In the same way as we proved Lemma 2.26 we then get that:

$$(\bar{\mathbf{x}}^\pi)^T \;=\; \mathbf{e}^T\sum_{t=0}^{\infty}(\gamma Q)^t \;=\; \mathbf{e}^T(I-\gamma Q)^{-1}\;.$$

Next we show that $\mathbf{x}^\pi = \mathbf{x}$. It suffices to show that $\bar{\mathbf{x}}^\pi = \bar{\mathbf{x}}$, since the definition of π then ensures that the flux is distributed correctly on the different actions. Since $\mathbf{x}$ is a feasible solution to (P) it must satisfy:

$$\forall i \in S: \quad \bar{\mathbf{x}}_i \;=\; \sum_{a\in A_i}\mathbf{x}_a \;=\; 1+\sum_{j\in S}\sum_{a\in A_j}\mathbf{x}_a\,\gamma P_{a,i} \;=\; 1+\sum_{j\in S}\bar{\mathbf{x}}_j\,\gamma Q_{j,i} \;=\; 1+\bar{\mathbf{x}}^T\gamma Q\mathbf{e}_i$$

Using matrix notation this equality says that $\bar{\mathbf{x}} = \mathbf{e} + \gamma Q^T\bar{\mathbf{x}}$, or $\bar{\mathbf{x}}^T = \mathbf{e}^T(I-\gamma Q)^{-1}$. Since $(\bar{\mathbf{x}}^\pi)^T = \mathbf{e}^T(I-\gamma Q)^{-1}$, we have shown that $\bar{\mathbf{x}}^\pi = \bar{\mathbf{x}}$ as desired. □

Theorem 2.30 *The linear program (P) has an optimal solution $\mathbf{x}$, and from $\mathbf{x}$ we get an optimal time-dependent policy π defined by*

$$\forall t\geq 0\;\forall i\in S\;\forall a\in A_i: \quad \Pr\left[\pi(i,t)=a\right] \;=\; \frac{\mathbf{x}_a}{\sum_{b\in A_i}\mathbf{x}_b}\;.$$

Proof: Since, by Lemma 2.28, every policy gives a feasible solution for (P) we know that (P) is feasible. Furtermore, since, by lemmas 2.29 and 2.27, every feasible solution of (P) gives a policy with the same sum of values we know that (P) is bounded. Hence, (P) has an optimal solution, and the corresponding policy π, constructed in the proof of Lemma 2.29, is optimal.

□

Note that basic feasible solutions of (P) correspond to positional policies. The existence of an optimal basic feasible solution for (P) therefore gives an alternative proof for the existence of an optimal positional policy.

Furthermore, it is not difficult to see that the reduced cost vector from Definition 2.20 corresponds exactly to the reduced costs that the simplex method operates with. In fact, this shows that the policy iteration algorithm can be viewed as a generalization of the simplex method where multiple pivots are performed in parallel.

3 Turn-based stochastic games

We next consider a generalization of MDPs known as *turn-based stochastic games* (TBSGs). TBSGs are a special case of Shapley's *stochastic games* [12], which were introduced in 1953. A TBSG is an MDP where the set of states have been assigned to two players; player 1, the *minimizer*, and player 2, the *maximizer*. Whenever the game is in some state the player controlling that state decides which action to use. The goal of the minimizer is to minimize the incurred costs according to some criterion, while the goal of the maximizer is to maximize the incurred costs. Hence, the game is a *zero-sum* game. In this section we restrict our attention to the discounted cost criterion.

Definition 3.1 (Turn-based stochastic game) *A* turn-based stochastic game *(TBSG) is a tuple $G = (S_1, S_2, A, s, c, p)$ such that $S_1 \cap S_2 = \emptyset$ and $(S_1 \cup S_2, A, s, c, p)$ is an MDP. We let $S = S_1 \cup S_2$. We again let A_i be the set of actions applicable from state i. We also let $A^j = \bigcup_{i \in S_j} A_i$ be the set of actions controlled by player j, for $j \in \{1, 2\}$.*

We use the same notation to describe TBSGs as we did for MDPs. In particular, probability matrices, cost vectors, and source matrices are defined analogously. Policies are also defined in the same way, except that a policy belongs to a player and only describes the decision made by that player. Also, in order to be consistent with terminology from game theory we use the term *strategy* instead of policy. We restrict our attention to positional strategies. As for MDPs it can be shown that the players are not worse off by being restricted to positional strategies. The proof is essentially the same for TBSGs.

Definition 3.2 (Strategies, strategy profiles) *A* positional strategy *π_j for player j, where $j \in \{1, 2\}$, is a mapping $\pi_j : S_j \to A$ such that $\pi_j(i) \in A_i$, for every $i \in S_j$. A* strategy profile *$\pi = (\pi_1, \pi_2)$ is a pair of strategies, one for each player. We denote the set of positional strategies for player j by $\Pi_j(\mathcal{P})$, and the set of strategy profiles by $\Pi(\mathcal{P}) = \Pi_1(\mathcal{P}) \times \Pi_2(\mathcal{P})$.*

Note that a strategy profile can be viewed as a policy for the underlying MDP. We again let P_π and $\mathbf{c}_\pi$ be obtained from P and $\mathbf{c}$ by selecting rows corresponding to actions used in π. Hence, the value for a state i when the players play according to a strategy profile π is again $\mathrm{val}_\pi^{D(\gamma)}(i) = \sum_{t=0}^{\infty} \gamma^t \mathrm{E}\left[c(Y_\pi(i,t))\right]$, and Lemma 2.3 shows that the value vector satisfies $\mathbf{v}^\pi = (I - \gamma P_\pi)^{-1}\mathbf{c}_\pi$.

Since there are two players with opposite objectives in a TBSG we need a different definition of optimal strategies compared to the definition of optimal policies for MDPs. Note, however, that if the strategy of one of the players is fixed then the game can be viewed as an MDP. Indeed, if we remove the unused actions of the player whose strategy is fixed, we may transfer control of his states to the other player. In this case the *optimal counter-strategy* of the player whose strategy is not fixed should be the optimal policy in the corresponding MDP. We get the following definition.

Definition 3.3 (Optimal counter-strategies) *Let G be a TBSG and let π_1 be a strategy for player 1. We say that a strategy π_2 for player 2 is an* optimal counter-strategy *against π_1 if and only if for all states $i \in S$:*

$$\mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i) \;=\; \max_{\pi_2' \in \Pi_2(\mathcal{P})} \mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2'}(i)\ .$$

Optimal counter-strategies are defined analogously for player 1 with max *exchanged with* min.

It follows from Theorem 1.6 that optimal counter-strategies always exist, since we get an MDP when the strategy of one player is fixed and MDPs always have optimal policies.

Let π_1 be some strategy. If π_2 is an optimal counter-strategy against π_1, then $\mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i)$ is the best possible value that player 2 can obtain for state i when player 1 uses the strategy π_1. Hence, we may view $\mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i)$ as an upper bound for the value player 1 can guarantee for state i. It is natural to ask what the best guarantee a player can get from a single strategy is. This leads to the following definition.

Definition 3.4 (Upper and lower values) *For every state $i \in S$ define the* upper value*, $\overline{\mathrm{val}}(i)$, and* lower value*, $\underline{\mathrm{val}}(i)$, by:*

$$\begin{aligned}
\overline{\mathrm{val}}(i) \;&=\; \min_{\pi_1 \in \Pi_1(\mathcal{P})} \; \max_{\pi_2 \in \Pi_2(\mathcal{P})} \; \mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i) \\
\underline{\mathrm{val}}(i) \;&=\; \max_{\pi_2 \in \Pi_2(\mathcal{P})} \; \min_{\pi_1 \in \Pi_1(\mathcal{P})} \; \mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i)
\end{aligned}$$

Note that we must have $\underline{\mathrm{val}}(i) \le \overline{\mathrm{val}}(i)$ for all states i. We say that a strategy is optimal when it achieves the best possible guarantee against the other player.

Definition 3.5 (Optimal strategies) *We say that a strategy π_1 for player 1 is* optimal *if and only if $\mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i) \le \overline{\mathrm{val}}(i)$ for all states i and strategies $\pi_2 \in \Pi_2(\mathcal{P})$. Similarly, a strategy π_2 for player 2 is* optimal *if and only if $\mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i) \ge \underline{\mathrm{val}}(i)$ for all i and $\pi_1 \in \Pi_1(\mathcal{P})$.*

The following theorem shows the existence of optimal strategies. The theorem was first established by Shapley [12]. It can be proved in essentially the same way as Theorem 1.6 for MDPs by using Theorem 3.8 below.

Theorem 3.6 *For any TBSG G there exists an optimal positional strategy profile (π_1, π_2). Moreover, for all states i we have:*

$$\mathrm{val}^{D(\gamma)}_{\pi_1,\pi_2}(i) \;=\; \overline{\mathrm{val}}(i) \;=\; \underline{\mathrm{val}}(i)\ .$$

By *solving* a TBSG we mean computing an optimal strategy profile. The following definition and theorem gives an alternative interpretation of optimal strategies. The theorem is standard for finite two-player zero-sum games. It was also established by Shapley [12]. We leave it as an exercise to prove the theorem.

Definition 3.7 (Nash equilibrium) *A strategy profile* $(\pi_1, \pi_2) \in \Pi$ *is a* Nash equilibrium *if and only if* π_1 *is an optimal counter-strategy against* π_2*, and* π_2 *is an optimal counter-strategy against* π_1*.*

Theorem 3.8

(i) If $(\pi_1, \pi_2) \in \Pi(\mathcal{P})$ *is optimal, then* (π_1, π_2) *is a Nash equilibrium.*

(ii) If $(\pi_1, \pi_2) \in \Pi(\mathcal{P})$ *is a Nash equilibrium then* $val^{D(\gamma)}_{\pi_1,\pi_2}(i) = \overline{\mathrm{val}}(i) = \underline{\mathrm{val}}(i)$ *for all* $i \in S$*, and* (π_1, π_2) *is optimal.*

3.1 The value iteration algorithm

Recall that a finite-horizon MDP runs for a fixed number of steps t, and that an additional cost $\gamma^t \mathbf{v}_i$ is added for terminating in state i, where $\mathbf{v} \in \mathbb{R}^n$ is some given vector. A finite-horizon TBSG can be defined analogously. As for MDPs, a finite-horizon discounted TBSG can be solved using dynamic programming. The value iteration operator can be generalized to TBSGs as follows.

Definition 3.9 (Value iteration operator) *Let* $\mathbf{v} \in \mathbb{R}^n$ *be an arbitrary vector. Define the* value iteration operator $\mathcal{T} : \mathbb{R}^n \to \mathbb{R}^n$ *by:*

$$\begin{aligned} \forall i \in S_1 : \quad (\mathcal{T}\mathbf{v})_i &= \min_{a \in A_i} \mathbf{c}_a + \gamma P_a \mathbf{v} \\ \forall i \in S_2 : \quad (\mathcal{T}\mathbf{v})_i &= \max_{a \in A_i} \mathbf{c}_a + \gamma P_a \mathbf{v} \; . \end{aligned}$$

Definition 3.10 (Strategy extraction operators) *The* strategy extraction operators $\mathcal{P}_1 : \mathbb{R}^n \to \Pi_1(\mathcal{P})$, $\mathcal{P}_2 : \mathbb{R}^n \to \Pi_2(\mathcal{P})$, *and* $\mathcal{P} : \mathbb{R}^n \to \Pi(\mathcal{P})$ *are defined by:*

$$\begin{aligned} \forall i \in S_1 : \quad (\mathcal{P}_1\mathbf{v})(i) &\in \operatorname*{argmin}_{a \in A_i} \mathbf{c}_a + \gamma P_a \mathbf{v} \; , \\ \forall i \in S_2 : \quad (\mathcal{P}_2\mathbf{v})(i) &\in \operatorname*{argmax}_{a \in A_i} \mathbf{c}_a + \gamma P_a \mathbf{v} \; , \end{aligned}$$

and $\mathcal{P}\mathbf{v} = (\mathcal{P}_1\mathbf{v}, \mathcal{P}_2\mathbf{v})$*.*

Note that $\mathcal{T}^t\mathbf{v}$ is the optimal value vector for the finite-horizon TBSG that terminates after t steps with termination costs specified by $\mathbf{v} \in \mathbb{R}^n$.

The value iteration algorithm can also be defined for TBSGs. In fact, the algorithm remains the same except that we now use the value iteration operator for two players. We refer to Figure 2 for a description of the value iteration algorithm. The lemmas and theorems in Section 2.1 can be extended to the two-player setting without much modication. We leave this as an exercise.

3.2 The strategy iteration algorithm

We will next see how Howard's policy iteration algorithm [7] for solving MDPs can be extended to 2TBSGs in a natural way. We refer to the resulting algorithm as the strategy iteration algorithm. The strategy iteration algorithm was described for TBSGs by Rao *et al.* [10].

For every strategy profile $\pi = (\pi_1, \pi_2)$, we define the vector of reduced costs $\bar{\mathbf{c}}^\pi$ as we did for MDPs: $\bar{\mathbf{c}}^\pi = \mathbf{c} - (J - \gamma P)\mathbf{v}^\pi$. We again use the reduced costs to define improving switches.

Definition 3.11 (Improving switches) *We say that an action $a \in A^1$ is an* improving switch *for player 1 with respect to a strategy profile π if and only if $\bar{\mathbf{c}}^\pi_a < 0$. Similarly, an action $a \in A^2$ is an* improving switch *for player 2 with respect to π if and only if $\bar{\mathbf{c}}^\pi_a > 0$.*

The following lemma is a direct consequence of lemmas 2.18 and 2.19, i.e., since a policy is optimal for an MDP if and only if there are no improving switches, we get a similar statement about optimal counter-strategies.

Lemma 3.12 *Let $\pi = (\pi_1, \pi_2)$ be a strategy profile. π_1 is an optimal counter-strategy against π_2 if and only if no action $a \in A^1$ is an improving switch for player 1 w.r.t. π. Similarly, π_2 is an optimal counter-strategy against π_1 if and only if player 2 has no improving switches.*

From the definition of Nash equilibria and Theorem 3.8 we get the following important corollary.

Corollary 3.13 *A strategy profile $\pi = (\pi_1, \pi_2)$ is a Nash equilibrium, and optimal, if and only if neither player has an improving switch with respect to π.*

The following lemma can be viewed as a generalization of Lemma 2.18 to TBSGs.

Lemma 3.14 *Let $\pi = (\pi_1, \pi_2)$ and $\pi' = (\pi_1', \pi_2')$ be two strategy profiles such that π_2 is an optimal counter-strategy against π_1, π_1' is obtained from π_1 by performing improving switches w.r.t. π, and π_2' is an optimal counter-strategy against π_1'. Then $\mathbf{v}^{\pi'} < \mathbf{v}^\pi$.*

Proof: Observe first that since π_1' is obtained from π_1 by performing improving switches w.r.t. π we have $(\bar{\mathbf{c}}^\pi)_{\pi_1'} < \mathbf{0}$. Furthermore, since π_2 is an optimal counter-strategy against π_1, we get from Lemma 3.12 that player 2 has no improving switches with respect to π. That is, $(\bar{\mathbf{c}}^\pi)_a \leq 0$ for all $a \in A^2$, and in particular $(\bar{\mathbf{c}}^\pi)_{\pi_2'} \leq \mathbf{0}$. Hence, $(\bar{\mathbf{c}}^\pi)_{\pi'} < \mathbf{0}$ which means that $\mathcal{T}_{\pi'}\mathbf{v}^\pi < \mathbf{v}^\pi$. It follows from Lemma 2.18 that $\mathbf{v}^{\pi'} < \mathbf{v}^\pi$. $\square$

The strategy iteration algorithm is given in Figure 4. It starts with some initial strategy profile $\pi^0 = (\pi_1^0, \pi_2^0)$ and generates a sequence $(\pi^k)_{k=0}^N$, where $\pi^k = (\pi_1^k, \pi_2^k)$, of strategy profiles, ending with an optimal strategy profile π^N. The algorithm repeatedly updates the

```
Function StrategyIteration($\pi_1, \pi_2$)
  while $\exists \pi'_2 : \mathcal{T}_{\pi_1,\pi'_2}\mathbf{v}^{\pi_1,\pi_2} > \mathbf{v}^{\pi_1,\pi_2}$ do
    $\pi_2 \leftarrow \pi'_2$;
  while $\exists \pi'_1 : \mathcal{T}_{\pi'_1,\pi_2}\mathbf{v}^{\pi_1,\pi_2} < \mathbf{v}^{\pi_1,\pi_2}$ do
    $\pi_1 \leftarrow \pi'_1$;
    while $\exists \pi'_2 : \mathcal{T}_{\pi_1,\pi'_2}\mathbf{v}^{\pi_1,\pi_2} > \mathbf{v}^{\pi_1,\pi_2}$ do
      $\pi_2 \leftarrow \pi'_2$;
  return $(\pi_1, \pi_2)$;
```

```
Function StrategyIteration($\pi_1, \pi_2$)
  while $\exists$ an improving switch for player 2 w.r.t. $(\pi_1, \pi_2)$ do
    Update $\pi_2$ by performing improving switches for player 2;
  while $\exists$ an improving switch for player 1 w.r.t. $(\pi_1, \pi_2)$ do
    Update $\pi_1$ by performing improving switches for player 1;
    while $\exists$ an improving switch for player 2 w.r.t. $(\pi_1, \pi_2)$ do
      Update $\pi_2$ by performing improving switches for player 2;
  return $(\pi_1, \pi_2)$;
```

Figure 4: Two equivalent formulations of the strategy iteration algorithm.

strategy π_2^k for player 2 by performing improving switches for player 2. Note that this can be viewed as running the policy iteration algorithm on the MDP obtained by fixing π_1^k, and therefore this inner loop always terminates. When π_2^k is an optimal counter-strategy against π_1^k, the algorithm updates π_1^k once by performing improving switches for player 1, and the process is restarted. If π_1^k can not be updated then neither of the players has an improving switch, and we know from Corollary 3.13 that $\pi^N = (\pi_1^N, \pi_2^N)$ is optimal. Furthermore, since π_2^k is an optimal counter-strategy against π_1^k, and π_1^{k+1} is obtained from π_1^k by performing improving switches with respect to (π_1^k, π_2^k), we get from Lemma 3.14 that $\mathbf{v}^{\pi^{k+1}} < \mathbf{v}^{\pi^k}$ for all $0 \le k < N$. It follows that the same strategy profile does not appear twice, and since there are only a finite number of strategy profiles the algorithm terminates. We get the following theorem. By an iteration we mean an iteration of the outer loop.

Theorem 3.15 *For every initial strategy profile (π_1, π_2),* StrategyIteration(π_1, π_2) *returns an optimal strategy profile after a finite number of iterations.*

It can be helpful to think of the strategy iteration algorithm as only operating with the strategy for player 1, and view the inner loop as a subroutine. Indeed, optimal counter-strategies for player 2 can also be found by other means, for instance by using linear programming. During each iteration of the strategy iteration algorithm the current values and reduced costs are computed, and an improvement rule decides which improving switches to perform. This is completely analogous to the policy iteration algorithm for MDPs. We will again say that Howard's improvement rule [7] picks the action from each state with most negative reduced cost, i.e., π_1^{k+1} satisfies:

$$\forall i \in S_1 : \quad \pi_1^{k+1}(i) \in \operatorname*{argmin}_{a \in A_i} \bar{\mathbf{c}}_a^{\pi^k} .$$

We refer to the resulting algorithm as Howard's strategy iteration algorithm, although Howard only introduced the algorithm for MDPs.

It is again possible to prove that the number of iterations required for Howard's strategy iteration algorithm to find an optimal strategy profile is at most the number of iterations required for the value iteration algorithm to do the same. This was proved in Lemma 2.22 for MDPs. We leave it as an exercise to provide a proof for TBSGs. Moreover, the bound obtained through the work of Ye [13]; Hansen, Miltersen, and Zwick [6]; and Scherrer [11] also holds for Howard's strategy iteration algorithm:

Theorem 3.16 *Starting with any strategy profile π, the number of iterations performed by Howard's strategy iteration algorithm is at most $O(\frac{m}{1-\gamma} \log \frac{1}{1-\gamma})$.*

There is no known way of formulating the solution of a TBSG as a linear program. In fact, it is a major open problem to find a polynomial time algorithm for solving TBSGs when γ is given as part of the input. When γ is fixed, Theorem 3.16 provides a strongly polynomial bound. On the other hand, it is easy to see that the problem of solving TBSGs is in both **NP** and co**NP**: an optimal strategy profile serves as a witness of the optimal value of a state being both above and below a given threshold.

4 The average cost criterion

In this section we give a very brief introduction to the average cost criterion. Proofs in this section have been omitted, and we refer to Puterman [9] for additional details. As it was the case for the discounted cost criterion, one can show that every MDP has an optimal positional policy for the average cost criterion. We restrict our attention to the use of positional policies. Throughout the section we let $M = (S, A, s, c, p)$ be some given MDP.

For the average cost criterion the value of a state i for a positional policy π is:

$$\mathrm{val}^A_\pi(i) \;=\; \mathrm{E}\left[\liminf_{N\to\infty} \frac{1}{N}\sum_{t=0}^{N-1} c(Y_\pi(i,t))\right] \;=\; \lim_{N\to\infty}\frac{1}{N}\sum_{t=0}^{N-1} \mathbf{e}_i^T P_\pi^t \mathbf{c}_\pi \;.$$

The fact that the limit exists is non-trivial.

Definition 4.1 (Values, potentials) *Let π be a positional policy, and let $\mathbf{g}^\pi \in \mathbb{R}^n$ and $\mathbf{h}^\pi \in \mathbb{R}^n$ be defined as:*

$$\begin{aligned}\mathbf{g}^\pi &= \lim_{N\to\infty}\frac{1}{N}\sum_{t=0}^{N-1} P_\pi^t \mathbf{c}_\pi \\ \mathbf{h}^\pi &= \lim_{N\to\infty}\frac{1}{N}\sum_{t=0}^{N-1}\sum_{k=0}^{t} P_\pi^k(\mathbf{c}_\pi - \mathbf{g}^\pi)\end{aligned}$$

We say that $\mathbf{g}^\pi$ and $\mathbf{h}^\pi$ are the vectors of values *and* potentials*, respectively.*

Note that $\mathrm{val}^A_\pi(i) = (\mathbf{g}^\pi)_i$ for every $i \in S$. If P_π is aperiodic, then $\mathbf{e}_i^T P_\pi^*$, where $P_\pi^* = \lim_{t\to\infty} P_\pi^t$, is the limiting, stationary distribution reached when starting in state i. In this case we have $(\mathbf{h}^\pi)_i = \sum_{t=0}^{\infty} \mathbf{e}_i^T P_\pi^t(\mathbf{c}_\pi - \mathbf{g}^\pi)$ and $(\mathbf{g}^\pi)_i = \mathbf{e}_i P_\pi^* \mathbf{c}_\pi$. Hence, $(\mathbf{h}^\pi)_i$ can be interpreted as the expected total difference in cost between starting in state i and starting according to $\mathbf{e}_i^T P_\pi^*$.

Recall that a recurrent class for a Markov chain is a set of states for which we can eventually get from every state to every other state with positive probability, and where the probability of leaving the set is zero.

Theorem 4.2 *Let π be a positional policy and let $\mathcal{R}^\pi \subseteq 2^S$ be the set of recurrent classes of P_π. Then $\mathbf{g}^\pi$ and $\mathbf{h}^\pi$ are the unique solution to the following system of equations:*

$$\begin{aligned}\mathbf{g}^\pi &= P_\pi \mathbf{g}^\pi \\ \mathbf{h}^\pi &= \mathbf{c}_\pi - \mathbf{g}^\pi + P_\pi \mathbf{h}^\pi \\ \forall R \in \mathcal{R}^\pi: \quad \sum_{i\in R} (\mathbf{h}^\pi)_i &= 0\;.\end{aligned}$$

We next establish a connection between values and potentials, and values for the corresponding discounted MDP. We let $\mathbf{v}^{\pi,\gamma}$ be the value vector for policy π for the discounted cost criterion when using discount factor γ.

Lemma 4.3 *For every positional policy π we have*

$$\mathbf{v}^{\pi,\gamma} \;=\; \frac{1}{1-\gamma}\mathbf{g}^{\pi} + \mathbf{h}^{\pi} + f^{\pi}(\gamma)$$

where $f^{\pi}(\gamma) \in \mathbb{R}^n$ is a vector, and $\lim_{\gamma \uparrow 1} f^{\pi}(\gamma) = \mathbf{0}$.

By multiplying both sides of the equation in Lemma 4.3 by $(1-\gamma)$, and taking the limit for γ going to 1 we get the following important Corollary:

Corollary 4.4 *Let π be any positional policy. Then:*

$$\mathbf{g}^{\pi} \;=\; \lim_{\gamma \uparrow 1}\, (1-\gamma)\mathbf{v}^{\pi,\gamma}\ .$$

Let $(\gamma_k)_{k=0}^{\infty}$ be a sequence of discount factors converging to 1. Since for every discount factor there is an optimal policy for the discounted MDP, and since there are only finitely many policies, there must be a policy that is optimal for discount factors arbitrarily close to 1. In fact, the following lemma shows something stronger.

Lemma 4.5 *There exists a discount factor $\gamma(M) < 1$ and a policy π^*, such that for all $\gamma \in [\gamma(M), 1)$, π^* is optimal for the discounted MDP with discount factor γ.*

By combining Lemma 4.5 and Corollary 4.4 it follows that π^* is optimal under the average cost criterion, which proves Theorem 1.5 for the average cost criterion. It should be noted that not every policy that is optimal under the average cost criterion is optimal under the discounted reward criterion for all discount factors sufficiently close to 1.

Lemmas 4.3 and 4.5 also show that we can use the strategy iteration algorithm for solving MDPs with the average cost criterion. Indeed, we can pick a discount factor γ sufficiently close to 1 and run the strategy iteration algorithm for the corresponding discounted MDP. To simplify the algorithm we can use Lemma 4.3 to compare reduced costs lexicographically in terms of values and potentials.

5 The total cost criterion

We next give a brief introduction to the total cost criterion. We again restrict our attention to positional policies. Throughout the section we let $M = (S, A, s, c, p)$ be some given MDP.

For the total cost criterion the value of a state i for a positional policy π is defined as:

$$\mathrm{val}_\pi^T(i) \;=\; \sum_{k=0}^{\infty} \mathbf{e}_i^T P_\pi^k \mathbf{c}_\pi$$

This series may not converge, and we therefore need to make assumptions about the MDP to ensure convergence. For this purpose we introduce the notion of a terminal state:

Definition 5.1 (Terminal state) *We say that a state* $\mathrm{T} \in S$ *is a terminal state if for all actions* $a \in A_{\mathrm{T}}$, $P_a = \mathbf{e}_{\mathrm{T}}$ *and* $\mathbf{c}_a = 0$. *I.e.,* T *is an absorbing state for every policy* π, *and once* T *is reached no additional cost is accumulated.*

For the remainder of this section we assume that the MDP under consideration has a terminal state T. Note that reaching the terminal state corresponds to terminating the MDP. Also, since the decision at T is fixed we will let T and its actions be implicit in the description of the MDP M. We therefore do not consider T to be an element of S, and we let P, J, and $\mathbf{c}$ be the matrices and vectors obtained by removing the column corresponding to T and the rows corresponding to A_{T}. I.e., rows of P may sum to less than one, and $1 - P_a\mathbf{e}$ is the probability of moving to T when using action a. Note that this provides a natural interpretation of the discounted cost criterion. The rows of the matrix γP sum to $\gamma < 1$, and for each action we move to the terminal state with probability $(1 - \gamma)$.

Definition 5.2 (Stopping condition) *A policy* π *is said to satisfy the* stopping condition *if from each state* i *there is positive probability of eventually reaching the terminal state. If every policy satisfies the stopping condition we say that the MDP satisfies the stopping condition.*

For the remainder of this section we assume that the MDP under consideration satisfies the stopping condition.

The following lemma is analogous to Lemma 2.1 for the discounted case.

Lemma 5.3 *For any policy* π *satisfying the stopping condition, the matrix* $(I - P_\pi)$ *is non-singular and*

$$(I - P_\pi)^{-1} \;=\; \sum_{k=0}^{\infty} P_\pi^k \;\geq\; I\ .$$

From Lemma 5.3 we get that values exist such that the following definition is well-defined.

Definition 5.4 (Value vector) *For every positional policy* π, *we define the* value vector $\mathbf{v}^\pi \in \mathbb{R}^n$ *by:*

$$\mathbf{v}^\pi = (I - P_\pi)^{-1}\mathbf{c}_\pi\ .$$

Note that if the MDP is deterministic, i.e., all actions move to single states with probability 1, then the problem becomes a shortest path problem, where from every state we want to find the shortest path to the terminal state. Solving MDPs for the total cost criterion may in general be viewed as a stochastic shortest path problem.

We next consider the relationship between the total cost criterion and the average cost criterion. For this purpose we assume that the terminal state T is represented explicitly. Note that, assuming that the MDP satisfies the stopping condition, the terminal state is reached from every state for every policy. Hence, the limiting average of the observed costs is always zero such that $\mathbf{g}^\pi = \mathbf{0}$ for every policy π. Furthermore, since the limit $\lim_{t\to\infty} \sum_{k=0}^{t} P_\pi^k \mathbf{c}_\pi$ always exists, we have:

$$\mathbf{h}^\pi \;=\; \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N-1} \sum_{k=0}^{t} P_\pi^k \mathbf{c}_\pi \;=\; \lim_{t\to\infty} \sum_{k=0}^{t} P_\pi^k \mathbf{c}_\pi \;=\; \mathbf{v}^\pi \;.$$

Hence, the values for the total cost criterion may be viewed as the potentials for the average cost criterion.

The existence of optimal positional policies can either be proved directly or through the connection to the average cost criterion. The strategy iteration algorithm can also be defined for the total cost criterion in the same way as it was done for the discounted cost criterion and the average cost criterion.

Through the series of papers by Friedmann [4] and Fearnley [3], from 2009 and 2010, respectively, it was shown that Howard's policy iteration algorithm may require exponentially many iterations in the number of states to solve MDPs with the total cost criterion. Hence, there is no hope of getting rid of the dependence on γ in Theorem 2.24.

Exercises

Exercise 1 Prove that for every MDP M, and every policy $\pi \in \Pi(\mathcal{H})$, the value $\mathrm{val}_\pi^{D(\gamma)}(i) = \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t c(Y_\pi(i,t))\right]$ always exists.

Exercise 2 Prove Lemma 2.1, i.e., for an MDP represented by $(P, \mathbf{c}, J)$ and any policy π, show that the matrix $(I - \gamma P_\pi)$ is nonsingular and that

$$(I - \gamma P_\pi)^{-1} \;=\; \sum_{k=0}^{\infty} (\gamma P_\pi)^k \;\geq\; I \;.$$

Hint: Consider the telescoping series $(I - \gamma P_\pi)\sum_{k=0}^{\infty}(\gamma P_\pi)^k$.

Exercise 3 Use Cramer's rule to prove Lemma 2.13. Cramer's rule says the following: Let $A\mathbf{x} = \mathbf{b}$ be a system of linear equations where $A \in \mathbb{R}^{n\times n}$ is a non-singular matrix and $\mathbf{b} \in \mathbb{R}^n$ is a vector. Then a solution $\mathbf{x}$ satisfies for all i, $\mathbf{x}_i = \det(A_i)/\det(A)$, where A_i is the matrix obtained from A by replacing the i-th column with $\mathbf{b}$.

Exercise 4 Give an alternative proof using complementary slackness for the fact that the linear programs (P) and (D) can be used to solve an MDP. The complementary slackness theorem says that $\mathbf{x}, \mathbf{y}, \mathbf{z}$ is an optimal solution for the primal and dual linear programs:

$$\begin{array}{lrcl} \min & \mathbf{c}^T\mathbf{x} & & \\ s.t. & A\mathbf{x} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array} \qquad \begin{array}{lrcl} \max & \mathbf{b}^T\mathbf{y} & & \\ s.t. & A^T\mathbf{y} + \mathbf{z} & = & \mathbf{c} \\ & \mathbf{z} & \geq & \mathbf{0} \end{array}$$

if and only if $\mathbf{x}, \mathbf{y}, \mathbf{z}$ is a feasible solution and $\mathbf{x}^T\mathbf{z} = 0$.

Exercise 5 Prove Theorem 3.8.

Exercise 6 Extend the lemmas and theorems in Section 2.1 to TBSGs and update the proofs. Hint: Lemma 2.7 should be split into two cases where different players use an optimal counter-strategy.

Exercise 7 Prove Lemma 2.22 in the setting of TBSGs.

References

[1] R. Bellman. *Dynamic programming.* Princeton University Press, 1957.

[2] F. d'Epenoux. A probabilistic production and inventory problem. *Management Science*, 10(1):98–108, 1963.

[3] J. Fearnley. Exponential lower bounds for policy iteration. In *Proc. of 37th ICALP*, pages 551–562, 2010.

[4] O. Friedmann. An exponential lower bound for the parity game strategy improvement algorithm as we know it. In *Proc. of 24th LICS*, pages 145–156, 2009.

[5] T. D. Hansen. *Worst-case Analysis of Strategy Iteration and the Simplex Method.* PhD thesis, Aarhus University, 2012.

[6] T. D. Hansen, P. B. Miltersen, and U. Zwick. Strategy iteration is strongly polynomial for 2-player turn-based stochastic games with a constant discount factor. *Journal of the ACM*, 60(1):1–16, 2013.

[7] R. Howard. *Dynamic programming and Markov processes.* MIT Press, 1960.

[8] U. Meister and U. Holzbaur. A polynomial time bound for Howard's policy improvement algorithm. *OR Spektrum*, 8:37–40, 1986.

[9] M. Puterman. *Markov decision processes.* Wiley, 1994.

[10] S. Rao, R. Chandrasekaran, and K. Nair. Algorithms for discounted games. *Journal of Optimization Theory and Applications*, pages 627–637, 1973.

[11] B. Scherrer. Improved and generalized upper bounds on the complexity of policy iteration. *CoRR*, abs/1306.0386, 2013.

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

[13] Y. Ye. The simplex and policy-iteration methods are strongly polynomial for the markov decision problem with a fixed discount rate. *Math. Oper. Res.*, 36(4):593–603, 2011.