
Algorithmic Game Theory and Applications

Lecture 15: a brief taster of Markov Decision Processes and Stochastic Games

Kousha Etessami

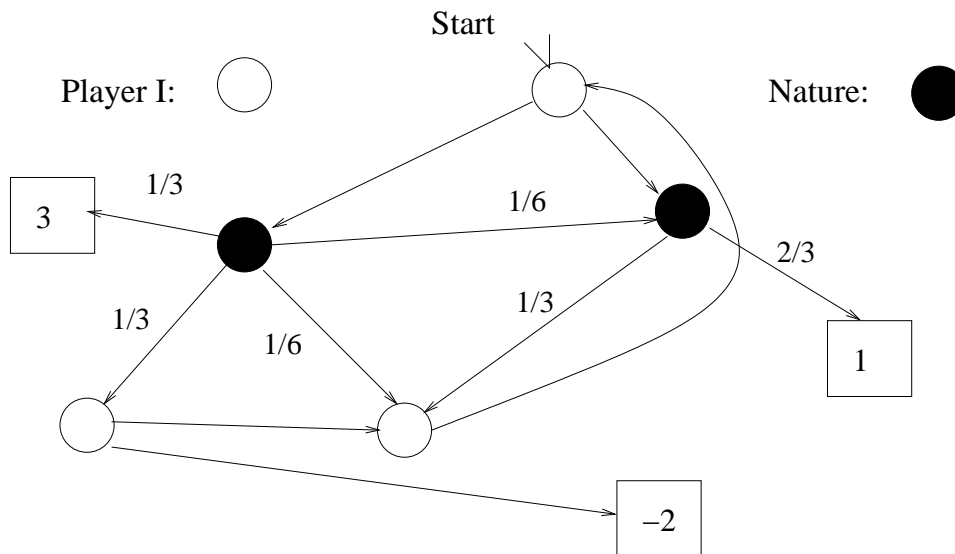


warning

- The subjects we will touch on today are so interesting and so vast that one could easily spend an entire course on them alone.
- So, think of what we discuss today as only a brief “taster”, and please do explore it further if it interests you.

Games against Nature

Consider a game graph, where some nodes belong to player 1 but others are chance nodes of “Nature”:



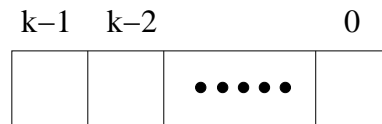
Question: What is Player 1’s “optimal strategy” and “optimal expected payoff” in this game?

We need to make sense of these questions formally.

But before doing that we’ll discuss applications.

(Note: in the above game every infinite play will infinitely often encounter a node of nature, and from each such node will “exit” with > 0 probability. So, here at least, “with probability 1” we will eventually exit and there is thus no need to talk about payoffs of “infinite” plays.)

a simple finite game: “make a big number”



- Your goal is to create as large a k -digit number as possible, using digits from $D = \{0, 1, 2, \dots, 9\}$, which “nature” will give you, one by one.
- The game proceeds in k rounds.
- In each round, “nature” chooses $d \in D$ “uniformly at random”, i.e., each digit has probability $1/10$.
- You then choose which “unfilled” position in the k -digit number should be “filled” with digit d . (Initially, all k positions are “unfilled”.)
- The game ends when all k positions are “filled”.

Your goal: maximize final number’s expected value.

Question: What should your strategy be?

This is a “finite horizon” “Markov Decision Process”. (example taken from [Puterman’94].)

Note that this is a finite PI-game and, in principle, we can solve it using the “bottom-up” algorithm.

But we wouldn’t want to look at the entire tree if we can avoid it!

vast applications

Beginning in the 1950's with the work of Bellman, Wald, and others, these kinds of "Games Against Nature", a.k.a., "Markov Decision Processes", a.k.a. "Stochastic Dynamic Programs", have been applied to a huge range of subjects.

Examples where MDPs have actually been applied:

- highway repair scheduling.
- inventory management.
- bus engine replacement.
- packet routing in telephone networks.
- waste management.
- call center scheduling (last week's LFCS seminar).
- mating and mate desertion in hawks (don't ask).
- hunting behavior of lions (don't ask).
- egg laying patterns of "apple magots" (don't ask).
- medical diagnosis and prediction (don't ask).
- (don't ask).

(See [Puterman'94, Ross'83, Derman'72, Howard'70, Bellman'57,..])

- The richness of applications shouldn't surprise you.
- We live in an uncertain world, where we constantly have to make decisions in the face of uncertainty about future events.
- But we may have some information, or "belief", about the "likelihood" of future events.
- "I know I may get hit by a car if I walk out of my apartment in the morning." But somehow I still muster the courage to get out.
- I don't however walk into a random pub in Glasgow and yell "I LOVE CELTIC FOOTBALL CLUB", because I know my chances of survival are far lower.

Markov Decision Processes: a restricted definition

We define restricted MDPs, sufficient for our purposes.

Definition A (restricted) Markov Decision Process is given by a game graph $G_{v_0} = (V, E, pl, q, v_0, u)$, where:

- V is a (finite) set of vertices.
- $pl : V \mapsto \{0, 1\}$, maps each vertex either to player 0 (“Nature”) or to player 1.
- Let $V_0 = pl^{-1}(0)$, and $V_1 = pl^{-1}(1)$.
- $E : V \mapsto 2^V$ maps each vertex v to a set $E(v)$ of “successors” (or “actions” at v).
- Let $V_{dead} = \{v \in V \mid E(v) = \emptyset\}$.
- For each “nature” vertex, $v \in V_0$, a probability distribution $q_v : E(v) \mapsto [0, 1]$, over the set of “actions” at v , such that $\sum_{v' \in E(v)} q_v(v') = 1$.
- A start vertex $v_0 \in V$.
- Here comes our first key restriction:

A finitistic history oblivious payoff function

$u : \Psi_{T_{v_0}} \mapsto \mathbb{R}$, from plays to payoffs to player 1.

Note: If the probability that play is infinite is 0, we can regard u as $u : V_{dead} \mapsto \mathbb{R}$. We want to somehow insure that this will be the case.....

terminating MDPs

Note that, as formulated, MDPs are extensive form PI-games. So, all the notions we developed for extensive games apply.

In particular, a (pure) “strategy” s_1 for player 1 in G_v is just the usual mapping from nodes in T_v belonging to player 1 to actions at those nodes.

Definition A MDP, G_{v_0} , is called **terminating** if for every $v \in V_0$, there exists $v' \in V_{dead}$ with $q_v(v') > 0$, and there are no cycles in the subgraph of G_{v_0} induced by V_1 .

We will only consider terminating MDPs.

Proposition For every terminating MDP, G_v , regardless of the strategy s_1 of player 1, the play reaches a dead end in V_{dead} “with probability 1”.

Obvious proof: Let

$$\lambda = \max_{v \in V_0} \left(1 - \sum_{v' \in (E(v) \cap V_{dead})} q_v(v') \right)$$

Note, $\lambda < 1$. The probability of reaching depth $m|V|$ in T_v without hitting a dead end is $\leq \lambda^m$. Thus, prob of infinite play is $\leq \lim_{m \rightarrow \infty} \lambda^m = 0$. ■

Thus, we only need to consider payoffs at dead ends.

Expected payoffs

- Intuitively, we want to define expected payoffs as the sum of payoffs of each (necessarily finite) play times its probability. Let $\Phi_{T_v} \subseteq \Psi_{T_v}$ denote the set of finite plays in T_v (which end at a dead end).
- For a tree node (i.e., word) $w \in T_{v_0}$, define $last(w)$ so that $last(w'v) := v$ and $last(\epsilon) := v_0$.
- Let $w' \sqsubseteq w$ denote that w' is a prefix of w .
- We call $w \in T_{v_0}$ consistent with strategy s_1 if for every prefix $w'v' \sqsubseteq w$, if $last(w') \in V_1$, then $s_1(w') = v'$. Let $C_{s_1} \subseteq \Phi_{T_v}$ denote the set of finite plays that are consistent with strategy s_1 .
- For $w \in C_{s_1}$, let

$$p(w) := \prod_{w'v' \sqsubseteq w \wedge last(w') \in V_0} q(last(w'))(v')$$

be the probability of play w , i.e., the product of probabilities of edges on the path traced by w .

- Define the expected payoff to player 1 as:

$$h(s_1) := \sum_{m=0}^{\infty} \sum_{w \in C_{s_1} \wedge |w|=m} p(w) * u(last(w))$$

This infinite series can be show to be convergent.

memoryless optimal strategies

How would we define “mixed” strategies $x_1 \in X_1$, as a “randomized combination” of pure strategies $s_1 \in S_1$?

This is subtle to define. One way is to define “behavioral” mixed strategies where each $x_1(wv)$ is a probability distribution on “actions” in $E(v)$. This leads to a definition of expected payoff $h(x_1)$, etc. It turns out that such an extension won't help:

call a (pure) strategy s_1^* for player 1 **optimal** if:

$$h(s_1^*) = \sup_{x_1 \in X_1} h(x_1)$$

we would then call $h(s_1^*)$ the optimal **value** of the MDP G_v .

Theorem (memoryless “determinacy”) In every terminating MDP player 1 has a pure memoryless optimal strategy.

In other words, player 1 has an optimal strategy where it just picks one edge from $E(v)$ for each vertex $v \in V_1$.

We already have all the ingredients to prove this, but we just don't have the time.

computing optimal memoryless strategies via LP

Consider the following LP, a modification of the LP for solving finitistic 2-player games.

Let $V = \{v_1, \dots, v_n\}$ be the vertices of G_v . We will have one LP variable x_i for each vertex $v_i \in V$.

Minimize $\sum_{i=1}^n x_i$

Subject to:

$$x_i \geq \min_{v \in V_{dead}} u(v), \text{ for } i = 1, \dots, n;$$

$$x_i = u(v_i), \text{ for each } v_i \in V_{dead};$$

$$x_i \geq x_j, \text{ for each } v_i \in V_1, \text{ and } v_j \in E(v_i);$$

$$x_i = \sum_{v_j \in E(v_i)} q_{v_i}(v_j) * x_j, \text{ for each } v_i \in V_0;$$

Theorem For $(x_1^*, \dots, x_n^*) \in \mathbb{R}^n$ an optimal solution to this LP (which must exist), each x_i^* is the optimal value for player 1 in the game G_{v_i} .

Again, we have the ingredients to prove this, but not the time. See [Derman'72, Ross'83, Courcoubetis-Yannakakis'90, Filar-Vrieze'97].

Question: The LP gives us the optimal value, but where's the optimal strategy?

extracting the optimal strategy

Suppose you have computed the optimal values x^* for each vertex.

Then it turns out the memoryless optimal strategy for player 1 can be defined by:

$$s_1^*(v_i) := v_k, \text{ such that } x_k^* = \max_{v_j \in E(v_i)} x_j^*.$$

(Note that there may be ties for the successor with maximum value. Ties can be resolved arbitrarily.)

Remark: Amazingly, another way to find an optimal strategy for player 1 is to solve the dual LP!

In other words, an optimal solution to the dual LP encodes an optimal strategy of player 1 in the MDP associated with the primal LP. And, furthermore, if you use Simplex, the optimal basic feasible solution to the dual will yield a pure strategy. Too bad we don't have time to see this. (see [Filar-Vrieze'97].)

Stochastic Games

What if we introduce a second player in the game against nature?

In 1953 L. Shapley, one of the major figures in game theory, introduced “stochastic games”, a general class of zero-sum, not-necessarily perfect info, two-player games which generalize MDPs. This was about the same time that Bellman and others were studying MDPs.

We will confine ourselves to a restricted zero-sum, terminating, perfect information, fragment of stochastic games.

These are called “simple stochastic games” by [Condon'92]. (Condon restricts to win-lose games, but she points out the generalization to zero-sum.)

How do we define these formally? We have already done all the work with our definition of MDPs.

simple stochastic games

Definition A zero-sum **simple stochastic game** is given by a game graph $G_{v_0} = (V, E, pl, q, v_0, u)$, where:

- V is a (finite) set of vertices.
- $pl : V \mapsto \{0, 1, 2\}$, maps each vertex to one of player 0 (“Nature”), player 1, or player 2.
- Let $V_0 = pl^{-1}(0)$, $V_1 = pl^{-1}(1)$, & $V_2 = pl^{-1}(2)$.
- $E : V \mapsto 2^V$ maps each vertex v to a set $E(v)$ of “successors” (or “actions” at v).
- Let $V_{dead} = \{v \in V \mid E(v) = \emptyset\}$.
- For each “nature” vertex, $v \in V_0$, a probability distribution $q_v : E(v) \mapsto [0, 1]$, over the set of “actions” at v , such that $\sum_{v' \in E(v)} q_v(v') = 1$.
- A start vertex $v_0 \in V$.
- A finitistic history oblivious payoff function $u : \Psi_{T_{v_0}} \mapsto \mathbb{R}$, from plays to payoffs to player 1.
- Moreover, we assume G_{v_0} is terminating, meaning for every $v \in V_0$, there exists $v' \in V_{dead}$ with $q_v(v') > 0$, and there are no cycles in the subgraph of G_{v_0} induced by $V_1 \cup V_2$.

memoryless determinacy

- Given strategies $s_1 \in S_1$ and $s_2 \in S_2$, we can similarly define C_{s_1, s_2} as the set of finite plays that are consistent with the two strategies.
- We can then similarly define the expected payoffs, $h(s_1, s_2)$.
- We as usual call the game **determined** if

$$\sup_{s_1 \in S_1} \inf_{s_2 \in S_2} h(s_1, s_2) = \inf_{s_2 \in S_2} \sup_{s_1 \in S_1} h(s_1, s_2)$$

We will call it **memorylessly determined** if, as usual, we can pick the respective optimal strategies to be memoryless.

Theorem([Condon'92]) Every simple stochastic game is memorylessly determined.

computing optimal strategies

- Memoryless determinacy immediately gives us one algorithm for computing optimal strategies:
 - “Guess” the strategy for one of the two players.
 - The “residual game” is a MDP.
Solve the corresponding LP.
- This gives a **NP** \cap **co-NP** procedure for solving simple stochastic games.
- [Hoffman-Karp’66] studied a “strategy improvement algorithm” for stochastic games based on LP, which can be adapted to simple stochastic games ([Condon’92]). Nobody knows how efficient this algorithm is.
- Is there a P-time algorithm for solving simple stochastic games? This is an open problem. It is equivalent to the following statement about computational complexity classes ([Condon’92]):
“Alternating-randomized logarithmic space is equivalent to deterministic polynomial time.”
- It turns out that solving parity games (Lecture 13) can be efficiently reduced to solving simple stochastic games (see, e.g., [Zwick-Paterson’96]).

food for thought

This food for thought is rather unusual, because I have NO IDEA what, if anything, the answer is. It is a mystery that I would like an answer to:

What, if anything, is the relationship between computing Nash Equilibria in finite (two-person) strategic games and computing solutions to (simple) stochastic games?

In other words:

What does Nash have to do with Shapley?

Of course, Nash and Shapley were friends in grad school when game theory was being founded, but that's obviously not what I mean!

(To put it more concretely: is either computational problem efficiently reducible to the other?)

UPDATE: we now know that both computing the value of Simple Stochastic Games, and approximating the (irrational) value of Shapley's Stochastic Games are reducible to computing a NE in 2-player strategic games. (see [Etes-Yann,FOCS'07]).)