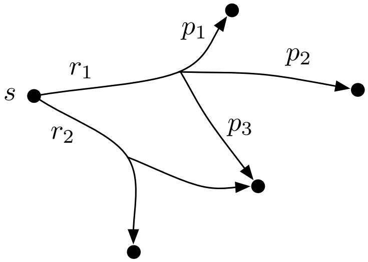
The simplex method is strongly polynomial for deterministic Markov decision processes

Ian Post Yinyu Ye

Fields Institute
November 29, 2013

Markov Decision Processes

A Markov decision process is a method of modeling repeated decision making over time in stochastic, changing environments.



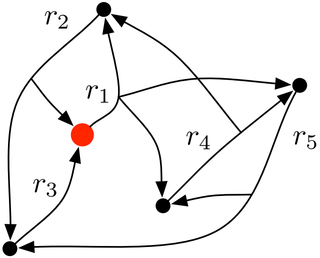
- It consists of states s and actions a with rewards r_a and probability distributions P_a over states
- When action a is used it receive the reward r_a and transitions to a new state according to the distribution P_a

Markov Decision Processes

- We are also given a discount factor $\gamma < 1$ as part of the input
- Goal: pick actions so as to maximize

$$\sum_{t=0}^{\infty} \gamma^t \mathbf{E}_{\mathcal{A}}[r(t)]$$

where $r(t)$ is the reward at time t



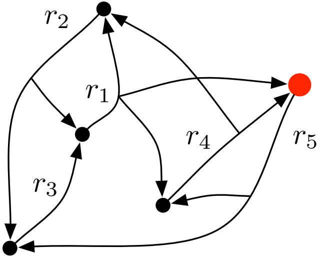
- Reward:

Markov Decision Processes

- We are also given a discount factor $\gamma < 1$ as part of the input
- Goal: pick actions so as to maximize

$$\sum_{t=0}^{\infty} \gamma^t \mathbf{E}_{\mathcal{A}}[r(t)]$$

where $r(t)$ is the reward at time t



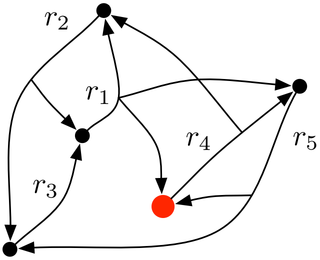
- Reward: r_1

Markov Decision Processes

- We are also given a discount factor $\gamma < 1$ as part of the input
- Goal: pick actions so as to maximize

$$\sum_{t=0}^{\infty} \gamma^t \mathbf{E}_{\mathcal{A}}[r(t)]$$

where $r(t)$ is the reward at time t



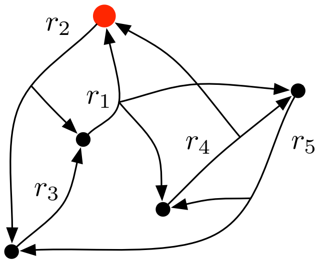
- Reward: $r_1 + \gamma r_5$

Markov Decision Processes

- We are also given a discount factor $\gamma < 1$ as part of the input
- Goal: pick actions so as to maximize

$$\sum_{t=0}^{\infty} \gamma^t \mathbf{E}_{\mathcal{A}}[r(t)]$$

where $r(t)$ is the reward at time t



- Reward: $r_1 + \gamma r_5 + \gamma^2 r_4$

Motivation

- MDPs are widely used in machine learning, operations research, economics, robotics and control, etc.

Motivation

- MDPs are widely used in machine learning, operations research, economics, robotics and control, etc.
- MDPs are also an interesting problem theoretically in that they are essentially where our knowledge of how to solve LPs in strongly polynomial time stops
  - Close to being strongly polynomial [Ye05] and possess a lot of structure that allows for powerful algorithms like policy iteration [How60]...
  - ...but also appear hard for powerful algorithms [Fea10] [FHZ11]

Motivation

- MDPs are widely used in machine learning, operations research, economics, robotics and control, etc.
- MDPs are also an interesting problem theoretically in that they are essentially where our knowledge of how to solve LPs in strongly polynomial time stops
  - Close to being strongly polynomial [Ye05] and possess a lot of structure that allows for powerful algorithms like policy iteration [How60]...
  - ...but also appear hard for powerful algorithms [Fea10] [FHZ11]
- Performance of basis exchange algorithms like policy iteration and simplex remains poorly understood
  - A number of open questions including their performance on special cases like deterministic MDPs [HZ10]
  - Important for developing new algorithms with better performance

Previous Work

- Policy iteration [How60]
  - Long conjectured to be strongly polynomial but only exponential bounds known [MS99]
  - Recently shown to be exponential [Fea10]
- Simplex lower bounds using MDPs [FHZ11] [Fri11] [MC94]
- Discounted MDPs (bounds depend on $\frac{1}{1-\gamma}$)
  - ϵ-approximation to the optimum [Bel57]
  - True optimum [Ye11] [HMZ11]
- Specialized algorithms for deterministic MDPs and other special cases [PT87] [HN94] [MTZ10] [Mad02]

Results

Theorem

The simplex method with Dantzig's most-negative reduced cost pivoting rule converges in $O(n^3 m^2 \log^2 n)$ iterations for deterministic MDPs regardless of the discount factor.

Theorem

If each action can have a distinct discount, then the simplex method converges in $O(n^5 m^3 \log^2 n)$ iterations.

Results

Theorem

The simplex method with Dantzig's most-negative reduced cost pivoting rule converges in $O(n^3 m^2 \log^2 n)$ iterations for deterministic MDPs regardless of the discount factor.

Theorem

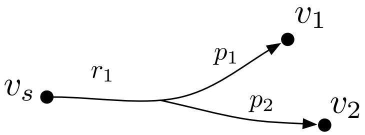
If each action can have a distinct discount, then the simplex method converges in $O(n^5 m^3 \log^2 n)$ iterations.

- Subsequent work [HKZ13] has improved these bounds by a factor of n

Value vector

- Let π be a *policy* (a choice of action for each state)
  - This defines a Markov chain
- The value (dual variable) $\mathbf{v}_s^\pi$ of a state s is the expected reward for starting in the state and following π

$$\mathbf{v}_s^\pi = \mathbf{r}_a + \gamma (P_a^\pi)^T \mathbf{v}^\pi$$

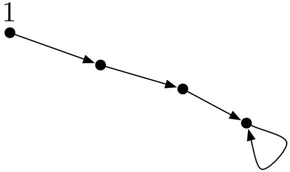


  - Key property: increasing the value of one state only increases values of others

Flux vector

- The flux (primal variable) $\mathbf{x}_a^\pi$ through an action a is the discounted number of times an action is used when starting in all the states

$$\mathbf{x}^\pi = \sum_{i \geq 0} (\gamma P^\pi)^i \mathbf{1} = (I - \gamma P^\pi)^{-1} \mathbf{1} \ ,$$

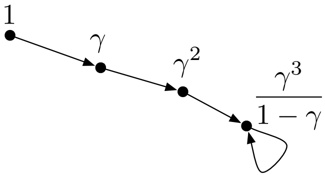


  - Flux through an action in π is always between 1 and $\frac{n}{1-\gamma} = n \sum_{i=0}^{\infty} \gamma^i$

Flux vector

- The flux (primal variable) $\mathbf{x}_a^\pi$ through an action a is the discounted number of times an action is used when starting in all the states

$$\mathbf{x}^\pi = \sum_{i \geq 0} (\gamma P^\pi)^i \mathbf{1} = (I - \gamma P^\pi)^{-1} \mathbf{1} \ ,$$



  - Flux through an action in π is always between 1 and $\frac{n}{1-\gamma} = n \sum_{i=0}^{\infty} \gamma^i$

Linear Program

MDPs can be solved with the following primal/dual pair of LPs

$$
\begin{array}{rll}
\textsc{Primal:} & & \\
\text{maximize} & \sum_a \mathbf{r}_a \mathbf{x}_a & \\
\text{subject to} & \forall s \in S, \quad \sum_{a \in A_s} \mathbf{x}_a & = 1 + \gamma \sum_a P_{a,s} \mathbf{x}_a \\
 & \mathbf{x} & \geq 0
\end{array}
$$

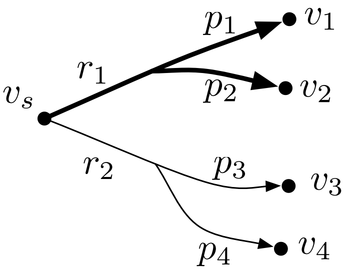
$$
\begin{array}{rl}
\textsc{Dual:} & \\
\text{minimize} & \sum_s \mathbf{v}_s \\
\text{subject to} & \forall s \in S, a \in A_s, \quad \mathbf{v}_s \geq \mathbf{r}_a + \gamma \sum_{s'} P_{a,s'} \mathbf{v}_{s'}
\end{array}
$$

Gain

- The gain (reduced cost) $\mathbf{r}_a^\pi$ of an action is improvement for switching to that action for one step

$$\mathbf{r}_a^\pi = (\mathbf{r}_a + \gamma P_a^T \mathbf{v}^\pi) - \mathbf{v}_s^\pi$$

- We will pivot on the action with the highest gain



$$r_1^\pi = (r_1 + \gamma(p_1 v_1 + p_2 v_2)) - v_s$$

Discounted MDPs

Basic idea: all variables lie in an interval of polynomial size. As a result the gap to the optimum shrinks by a polynomial factor each iteration.

- Suppose $\frac{1}{1-\gamma}$ is polynomial.
- Let π be the current policy and $\Delta = \max \mathbf{r}^\pi$ and $a = \operatorname{argmax} \mathbf{r}^\pi$
- $\mathbf{r}^T\mathbf{x}^* - \mathbf{r}^T\mathbf{x}^\pi = (\mathbf{r}^\pi)^T\mathbf{x}^* \leq \Delta\frac{n}{1-\gamma}$
- Using action a will increase objective by at least Δ, so distance to optimum shrinks by factor of $1 - \frac{1-\gamma}{n}$

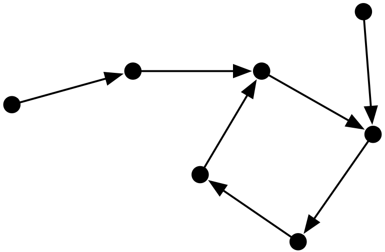
Discounted MDPs

- Now consider optimal gains $\mathbf{r}^*$
- Suppose $\Delta = \min_{a' \in \pi} \mathbf{r}^*$ and $a = \operatorname{argmin}_{a' \in \pi} \mathbf{r}^*$
- $\Delta > \mathbf{r}^T \mathbf{x}^\pi - \mathbf{r}^T \mathbf{x}^* > \Delta \frac{n}{1-\gamma}$ if $a \in \pi$.
- Therefore if $\mathbf{r}^T \mathbf{x}^\pi - \mathbf{r}^T \mathbf{x}^*$ shrinks by factor of $\frac{n}{1-\gamma}$, a can never again appear in a policy, and this happens after

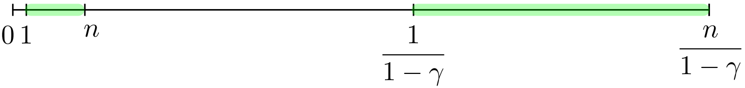
$$\log_{1-(1-\gamma)/n} \frac{1-\gamma}{n} = O\left(\frac{n}{1-\gamma} \log \frac{n}{1-\gamma}\right)$$

rounds [Ye10]

Deterministic MDPs



- An action is either on a path or a cycle
- If a is on a path then $\mathbf{x}_a \in [1, n]$
- If a is on a cycle then $\mathbf{x}_a \in \left[\frac{1}{1-\gamma}, \frac{n}{1-\gamma}\right]$
- So if $\mathbf{x}_a \neq 0$, it must lie in one of two *layers* of polynomial size



Uniform discount

Lemma

If the algorithm updates a path action it reduces the gap to the last policy before creating a new cycle by a factor of $1 - 1/n^2$.

Uniform discount

Lemma

If the algorithm updates a path action it reduces the gap to the last policy before creating a new cycle by a factor of $1 - 1/n^2$.

Lemma

After $O(n^2 \log n)$ *iterations, either the algorithm finishes, creates a new cycle, breaks a cycle, or some action never again appears in a policy before a new cycle is created.*

Uniform discount

Lemma

If the algorithm updates a path action it reduces the gap to the last policy before creating a new cycle by a factor of $1 - 1/n^2$.

Lemma

After $O(n^2 \log n)$ *iterations, either the algorithm finishes, creates a new cycle, breaks a cycle, or some action never again appears in a policy before a new cycle is created.*

Lemma

After $O(n^2 m \log n)$ *iterations, either the algorithm finishes or creates a new cycle.*

Uniform discount

Lemma

If the algorithm creates a new cycle it reduces the gap to the optimum by a factor of $1 - 1/n$.

Lemma

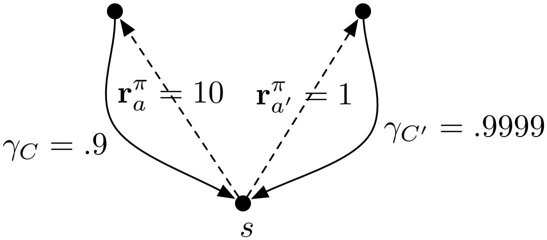
After $O(n \log n)$ *cycles are created either the algorithm finishes, some action is eliminated from cycles for the remainder of the algorithm or entirely eliminated from future policies, or the algorithm converges.*

Theorem

The simplex method converges in $O(n^3 m^2 \log^2 n)$ *iterations on deterministic MDPs.*

Nonuniform discount

- Now each action a has its own discount γ_a
- Problem: no more conservation of flux!
- Previously used highest gain to bound distance to the optimum, but now this is no longer possible

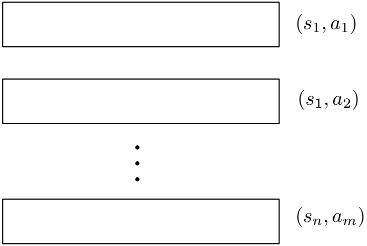


- Different cycles in a policy may have vastly different amounts of flux

Nonuniform discount

Basic idea:

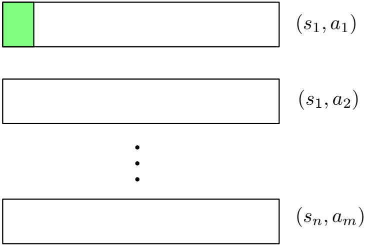
- The discount/flux in a cycle roughly determined by lowest discount action on the cycle
- When a cycle is created we lot of progress towards the optimal value of some state, *assuming its optimal flux is in that range*



Nonuniform discount

Basic idea:

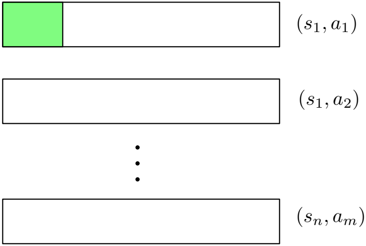
- The discount/flux in a cycle roughly determined by lowest discount action on the cycle
- When a cycle is created we lot of progress towards the optimal value of some state, *assuming its optimal flux is in that range*



Nonuniform discount

Basic idea:

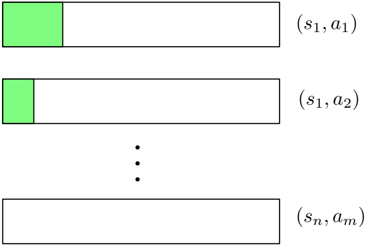
- The discount/flux in a cycle roughly determined by lowest discount action on the cycle
- When a cycle is created we lot of progress towards the optimal value of some state, *assuming its optimal flux is in that range*



Nonuniform discount

Basic idea:

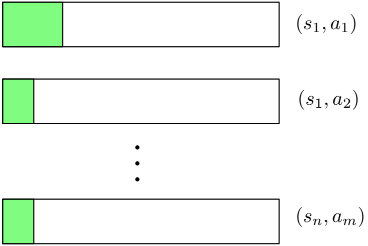
- The discount/flux in a cycle roughly determined by lowest discount action on the cycle
- When a cycle is created we lot of progress towards the optimal value of some state, *assuming its optimal flux is in that range*



Nonuniform discount

Basic idea:

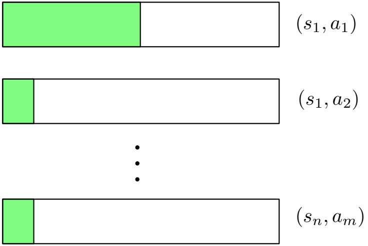
- The discount/flux in a cycle roughly determined by lowest discount action on the cycle
- When a cycle is created we lot of progress towards the optimal value of some state, *assuming its optimal flux is in that range*



Nonuniform discount

Basic idea:

- The discount/flux in a cycle roughly determined by lowest discount action on the cycle
- When a cycle is created we lot of progress towards the optimal value of some state, *assuming its optimal flux is in that range*



Theorem

The algorithm terminates in $O(n^5 m^3 \log^2 n)$ *iterations.*

Open questions

- Analyze policy iteration on deterministic MDPs
- Strongly polynomial algorithm for MDPs
- Apply layer idea to other problems