Efficiency Analysis of the Simplex Method

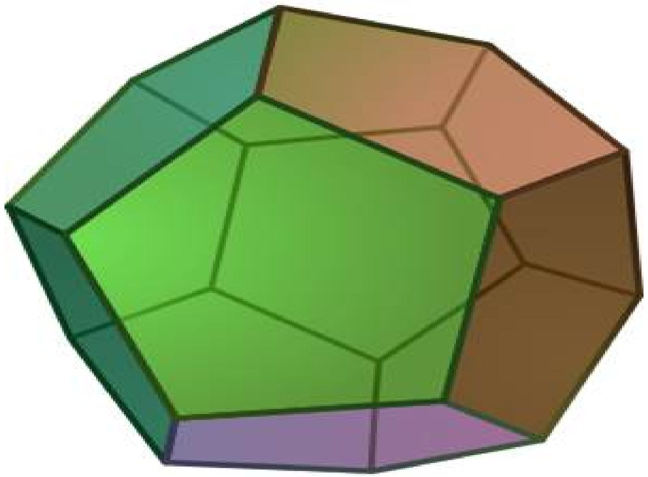
Yinyu Ye
Department of Management Science and Engineering
Stanford University
Stanford, CA 94305, U.S.A.

http://www.stanford.edu/~yyye

Chapter 4.6, 12.10, Wikipedia, Google on MDP

Hirsch's Conjecture

Warren Hirsch conjectured in 1957 that the diameter of the graph of a (convex) polyhedron defined by n inequalities in m dimensions is at most $n - m$. The diameter of the graph is the maximum of the shortest paths between every two vertices.



Counter Examples:

- Francisco Santos (2010): there is a 43-dimensional polytope with 86 facets and of diameter at least 44.
- There is an infinite family of non-Hirsch polytopes with diameter $(1 + \epsilon)n$, even in fixed dimension.

Size of Basic Feasible Solution and Convergence Rate

The simplex method generates a sequence of BFS $\{\mathbf{x}^k\}_{k=0,1,...}$ where the objective value decreases in each step, i.e., $\mathbf{c}^T\mathbf{x}^{k+1} \leq \mathbf{c}^T\mathbf{x}^k$.

Lemma 1 *For every BFS, say $\mathbf{x}_B$, of a LP problem, assume that the sum of its entries is bounded above*

$$\mathbf{e}^T\mathbf{x}_B \leq \Delta,$$

and its smallest entry is bounded below

$$\min\{\mathbf{x}_B\} \geq \delta > 0$$

for some positive constants Δ and δ (non-degenerate case). Then in every pivot step, we have

$$\frac{\mathbf{c}^T\mathbf{x}^{k+1} - z^*}{\mathbf{c}^T\mathbf{x}^k - z^*} \leq 1 - \frac{\delta}{\Delta}$$

where z^* is the minimal objective value of the LP problem.

Proof of the Convergence Rate

Recall at each pivot step,

$$r_e^k = \min_{j \in N}\{r_j^k\} < 0$$

where $\mathbf{r}^k = \mathbf{c} - A^T\mathbf{y}^k$ and $\mathbf{y}^k$ is the shadow price vector at the kth step. Thus,

$$\mathbf{c}^T\mathbf{x}^k - z^* = \mathbf{c}^T\mathbf{x}^k - \mathbf{c}^T\mathbf{x}^* = (\mathbf{r}^k)^T\mathbf{x}^k - (\mathbf{r}^k)^T\mathbf{x}^* = -(\mathbf{r}^k)^T\mathbf{x}^* \le -r_e^k \cdot \Delta.$$

On the other hand, we have

$$\mathbf{c}^T\mathbf{x}^{k+1} - \mathbf{c}^T\mathbf{x}^k = (\mathbf{r}^k)^T\mathbf{x}^{k+1} - (\mathbf{r}^k)^T\mathbf{x}^k = (\mathbf{r}^k)^T\mathbf{x}^{k+1} = r_e^k \cdot x_e^{k+1} \le r_e^k \cdot \delta.$$

Thus

$$(\mathbf{c}^T\mathbf{x}^{k+1} - z^*) - (\mathbf{c}^T\mathbf{x}^k - z^*) \le r_e \cdot \delta$$

or

$$\frac{\mathbf{c}^T\mathbf{x}^{k+1} - z^*}{\mathbf{c}^T\mathbf{x}^k - z^*} \le 1 + \frac{r_e \cdot \delta}{\mathbf{c}^T\mathbf{x}^k - z^*} \le 1 - \frac{\delta}{\Delta}.$$

Implicit Elimination Theorem

Theorem 1 *Let* $\mathbf{x}^0$ *be any given BFS. Then there is an optimal nonbasic variable* $j^0 \in B^0$ *and* $j^0 \notin B^*$*, that would never appear in any of the BFSs generated by the simplex method after* $K := \lceil \frac{\Delta}{\delta} \cdot \log\left(\frac{m\Delta}{\delta}\right) \rceil$ *steps starting from* $\mathbf{x}^0$*.*

Then we have

Corollary 1 *For every BFS, say* $\mathbf{x}_B$*, of a LP problem, let the sum of its entries be bounded above*

$$\mathbf{e}^T \mathbf{x}_B \le \Delta,$$

and its smallest entry be bounded below

$$\min\{\mathbf{x}_B\} \ge \delta > 0$$

for some positive constants Δ *and* δ*. Then the Simplex method terminates in at most* $\lceil \frac{(n-m)\Delta}{\delta} \cdot \log\left(\frac{m\Delta}{\delta}\right) \rceil$ *steps.*

Proof of the Theorem

If the initial BFS $\mathbf{x}^0$ is not optimal, then we have

$$(\mathbf{r}^*)^T\mathbf{x}^0 = \mathbf{c}^T\mathbf{x}^0 - z^* > 0.$$

Then there must be some index $j^0 \in B^0$ and $j^0 \notin B^*$ such that

$$r^*_{j^0} x^0_{j^0} \geq \frac{\mathbf{c}^T\mathbf{x}^0 - z^*}{m},$$

or

$$r^*_{j^0} \geq \frac{\mathbf{c}^T\mathbf{x}^0 - z^*}{m\Delta}.$$

After $K = \lceil \frac{\Delta}{\delta} \cdot \log\left(\frac{m\Delta}{\delta}\right) \rceil$ steps starting from $\mathbf{x}^0$, from the lemma we must have

$$\mathbf{c}^T\mathbf{x}^K - z^* < \frac{\delta}{m\Delta}(\mathbf{c}^T\mathbf{x}^0 - z^*)$$

and it holds for all subsequent BFSs.

Suppose $j^0 \in B^K$, we have

$$r^*_{j^0} x^K_{j^0} \leq (\mathbf{r}^*)^T \mathbf{x}^K = \mathbf{c}^T \mathbf{x}^K - z^* < \frac{\delta}{m\Delta}(\mathbf{c}^T \mathbf{x}^0 - z^*)$$

or

$$r^*_{j^0} < \frac{\mathbf{c}^T \mathbf{x}^0 - z^*}{m\Delta}$$

which gives a contradiction.

Therefore, $j^0 \notin B^k$ for all $k = K, K+1, ...$ and it is implicitly eliminated for the rest of Simplex method consideration.

Recall RL and Markov Decision Process

- Reinforced Learning (RL) and Markov Decision Process (MDP) provide a mathematical framework for modeling sequential decision-making in situations where outcomes are partly random and partly under the control of a decision maker. They are useful for studying a wide range of optimization problems solved via Dynamic Programming (DP), where it was known at least as early as the 1950s (cf. Shapley 1953, Bellman 1957).

- At each time step, the process is in some state $i \in \{1, ..., m\}$ and the decision maker chooses an action $j \in \mathcal{A}_i$ that is available in state i. The process responds at the next time step by randomly moving into a new state i', and giving the decision maker a corresponding cost c_j.

- The probability that the process changes from i to i' is influenced by the chosen action j in state i. Specifically, it is given by the state transition function $\mathbf{p}_j$. But when take action $j \in \mathcal{A}_i$, the probability is conditionally independent of all previous states and actions. In other words, the state transitions of an MDP possess the Markov Property.

MDP Stationary Policy

- By a Stationary Policy for the decision maker, we mean a function $\pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ that specifies an action $\pi_i \in \mathcal{A}_i$ that the decision maker will choose for each state i.
- The min-present cost MDP is to find a stationary policy to minimize the expected discounted sum over an infinite horizon:

$$\sum_{t=0}^{\infty} \gamma^t E[c^{\pi_{i^t}}(i^t, i^{t+1})],$$

where $0 \leq \gamma < 1$ is a discount rate. Typically, we use $\gamma = \frac{1}{1+\rho}$ where ρ is the interest rate.

- Each stationary policy induces a Cost-to-Go value, y_i, for each state, and the optimal one meets the Bellman Principle:

$$y_i^* = \min_{j \in \mathcal{A}_i} \{\mathbf{c}_j + \gamma \mathbf{p}_j^T \mathbf{y}^*\}, \ \forall i.$$

Algorithmic Events of the MDP Methods I

- Shapley (1953) and Bellman (1957) developed a method called the Value-Iteration (VI) method to approximate the optimal state values.
- Another best known method is due to Howard (1960) and is known as the Policy-Iteration (PI) method, which generate an optimal policy in finite number of iterations in a distributed and decentralized way.
- de Ghellinck (1960), D'Epenoux (1960) and Manne (1960) showed that the MDP has an LP representation, so that it can be solved by the Simplex method of Dantzig (1947) in finite number of steps, and the Ellipsoid method of Kachiyan (1979) in polynomial time.

The Value-Iteration for MDP

$$
\begin{cases}
y_1 &= \min_{j \in \mathcal{A}_1} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\} \\
&\vdots \\
y_i &= \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\} \\
&\vdots \\
y_m &= \min_{j \in \mathcal{A}_m} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\},
\end{cases}
$$

where $\mathcal{A}_i$ represents all actions available in state i, and $\mathbf{p}_j$ is the state transition probabilities from state i to all states when action jth in state i is taken.

The Equivalent (Dual) LP Form of the MDP:

$$
\begin{array}{lrcl}
\text{maximize}_{\mathbf{y}} & \sum_{i=1}^{m} y_i & & \\
\text{subject to} & y_1 - \gamma \mathbf{p}_j^T \mathbf{y} & \leq & c_j,\ j \in \mathcal{A}_1 \\
 & & \vdots & \\
 & y_i - \gamma \mathbf{p}_j^T \mathbf{y} & \leq & c_j,\ j \in \mathcal{A}_i \\
 & & \vdots & \\
 & y_m - \gamma \mathbf{p}_j^T \mathbf{y} & \leq & c_j,\ j \in \mathcal{A}_m.
\end{array}
$$

The MDP-LP Primal Formulation

$$\begin{array}{lccccc}
\min_{\mathbf{x}} & \sum_{j\in\mathcal{A}_1} c_j x_j + & \ldots & + \sum_{j\in\mathcal{A}_m} c_j x_j & & \\
\text{s.t.} & \sum_{j\in\mathcal{A}_1} (\mathbf{e}_1 - \gamma \mathbf{p}_j) x_j + & \ldots & + \sum_{j\in\mathcal{A}_m} (\mathbf{e}_m - \gamma \mathbf{p}_j) x_j & = & \mathbf{e}, \\
 & \ldots & x_j & \ldots & \geq & 0,\ \forall j,
\end{array}$$

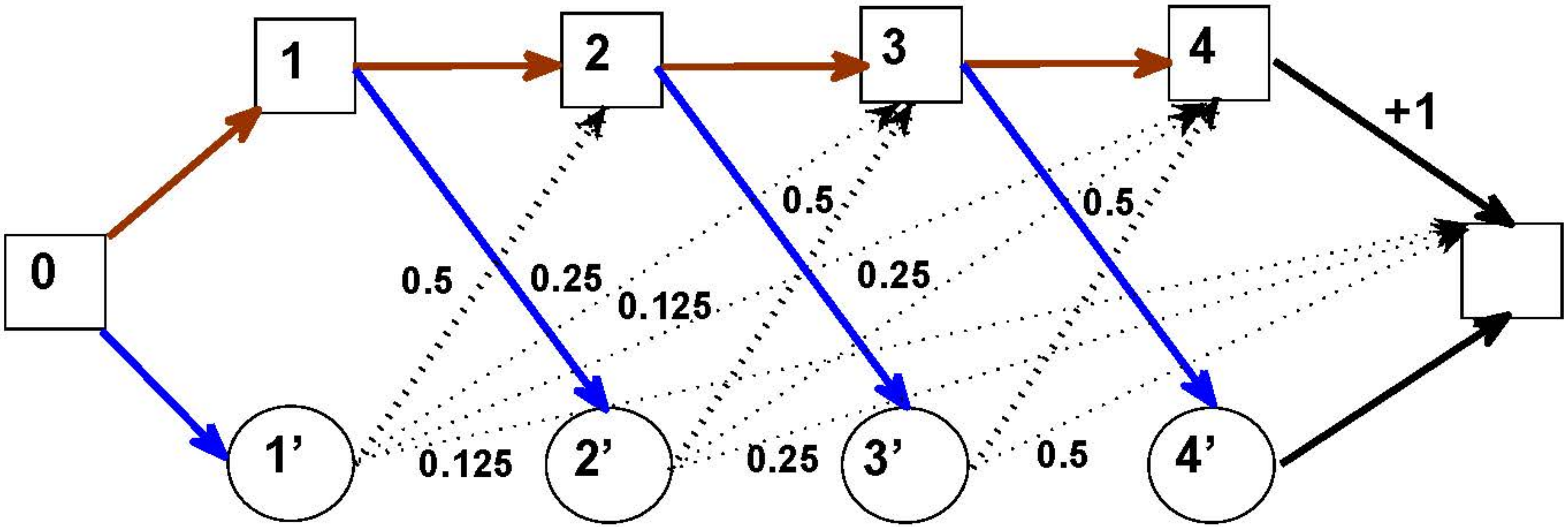
where $\mathbf{e}$ is the vector of ones, and $\mathbf{e}_i$ is the unit vector with 1 at the i-th position.

- Variable x_j, $j \in \mathcal{A}_i$, is the state-action frequency or flux, or the expected present value of the number of times in which an individual is in state i and takes state-action j. Thus, solving the problem entails choosing state-action frequencies/fluxes that minimize the expected present value sum of total costs.
- There is one-one correspondence between a stationary-policy and a BFS.
- When the Simplex Method is applied to solving the problem, the BFS update of becomes policy-update, and called Policy-Iteration method.

The Maze-Run Example

x:	(0_1)	(0_2)	(1_1)	(1_2)	(2_1)	(2_2)	(3_1)	(3_2)	(4_1)	(5_1)	**b**
c:	0	0	0	0	0	0	0	0	1	0	
(0)	1	1	0	0	0	0	0	0	0	0	1
(1)	$-\gamma$	0	1	1	0	0	0	0	0	0	1
(2)	0	$-\gamma/2$	$-\gamma$	0	1	1	0	0	0	0	1
(3)	0	$-\gamma/4$	0	$-\gamma/2$	$-\gamma$	0	1	1	0	0	1
(4)	0	$-\gamma/8$	0	$-\gamma/4$	0	$-\gamma/2$	$-\gamma$	0	1	0	1
(5)	0	$-\gamma/8$	0	$-\gamma/4$	0	$-\gamma/2$	0	$-\gamma$	$-\gamma$	$1-\gamma$	1

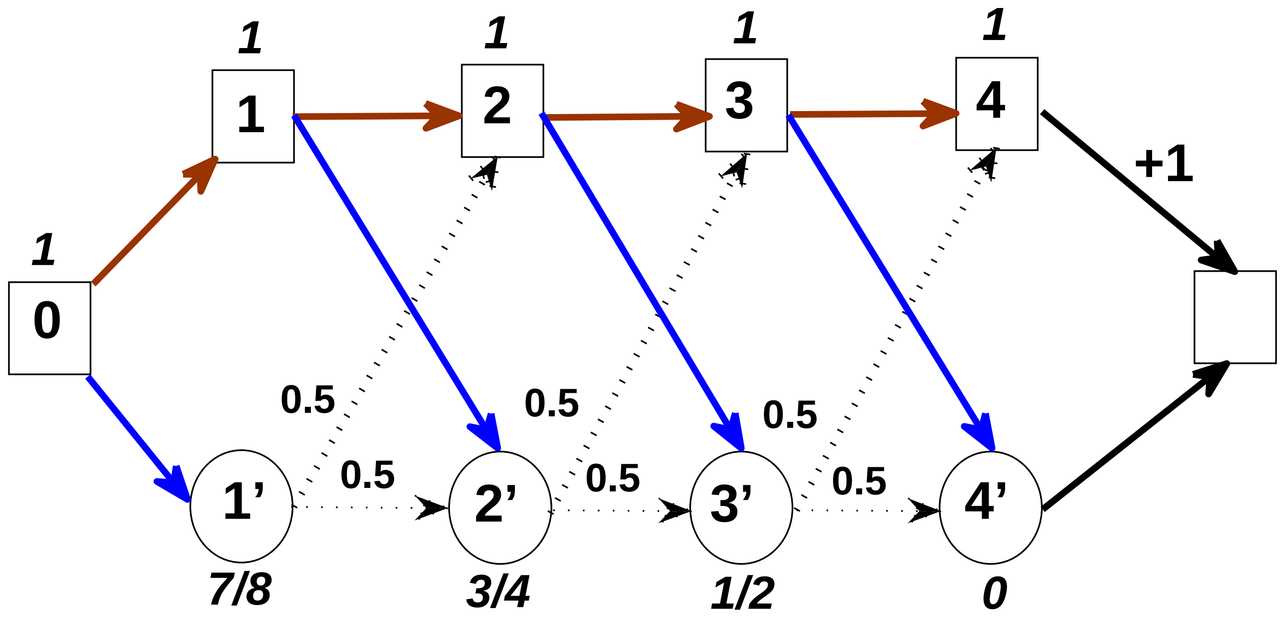
where state 5 is the absorbing state that has an infinite action-loops to itself.



The optimal fluxes are

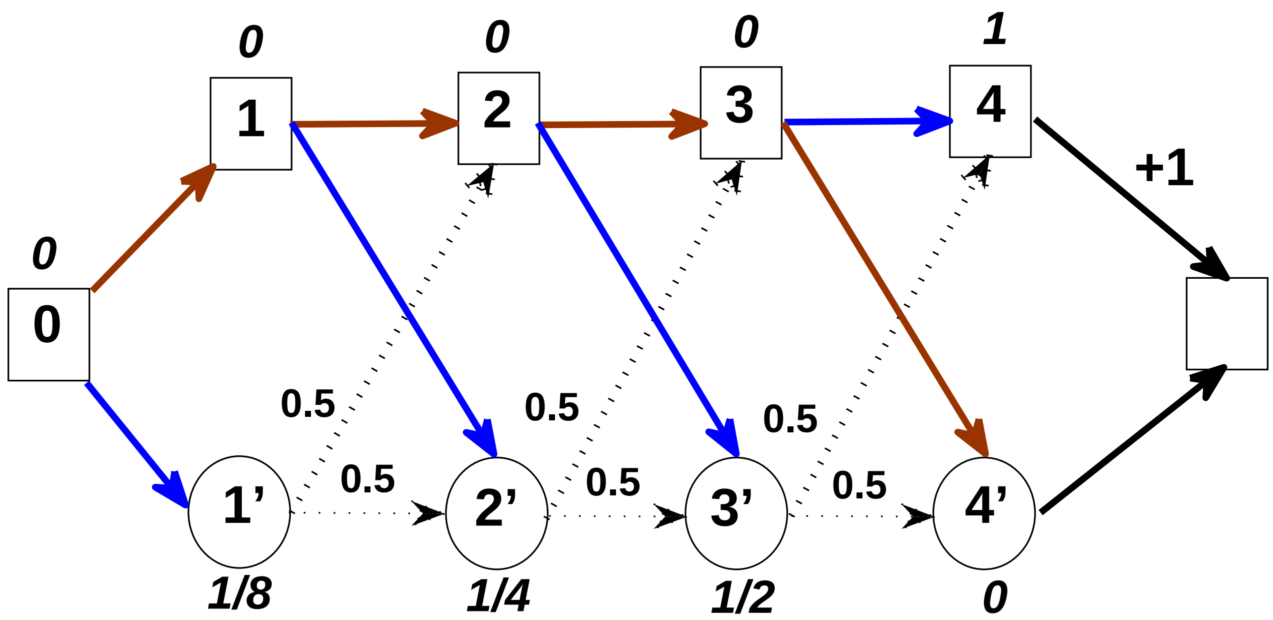
$$x^*_{01} = 1,\ x^*_{11} = 1 + \gamma,\ x^*_{21} = 1 + \gamma + \gamma^2,\ x^*_{32} = 1 + \gamma + \gamma^2 + \gamma^3,\ x^*_{41} = 1,\ x^*_{51} = \frac{1 + \gamma \cdot x^*_{32}}{1 - \gamma}.$$

The Policy-Iteration for MDP



The Cost-to-Go (or Dual) Values for each state when actions colored in red are taken or the initial BFS is $(x_{01}, x_{11}, x_{21}, x_{1}, x_{41}, x_{51})$.

The Simplex or Simple Policy-Iteration: greedy rule



The Simplex or Simple Policy Greedy-Rule Iteration: switch one action with the largest improvement rate among all states; new dual values on each state when actions in red are taken.

The (Classic) Policy Iteration Method for MDP

0. Initialize Start any policy or BFS with basic index set B. Let N denote the complementary index set.

1. Test for termination: Compute $\mathbf{x}_B = (A_B)^{-1}\mathbf{e} \geq \mathbf{0}$, $\mathbf{y}^T = \mathbf{c}_B^T(A_B)^{-1}$, and $\mathbf{r} = \mathbf{c} - A^T\mathbf{y}$.

2. Select

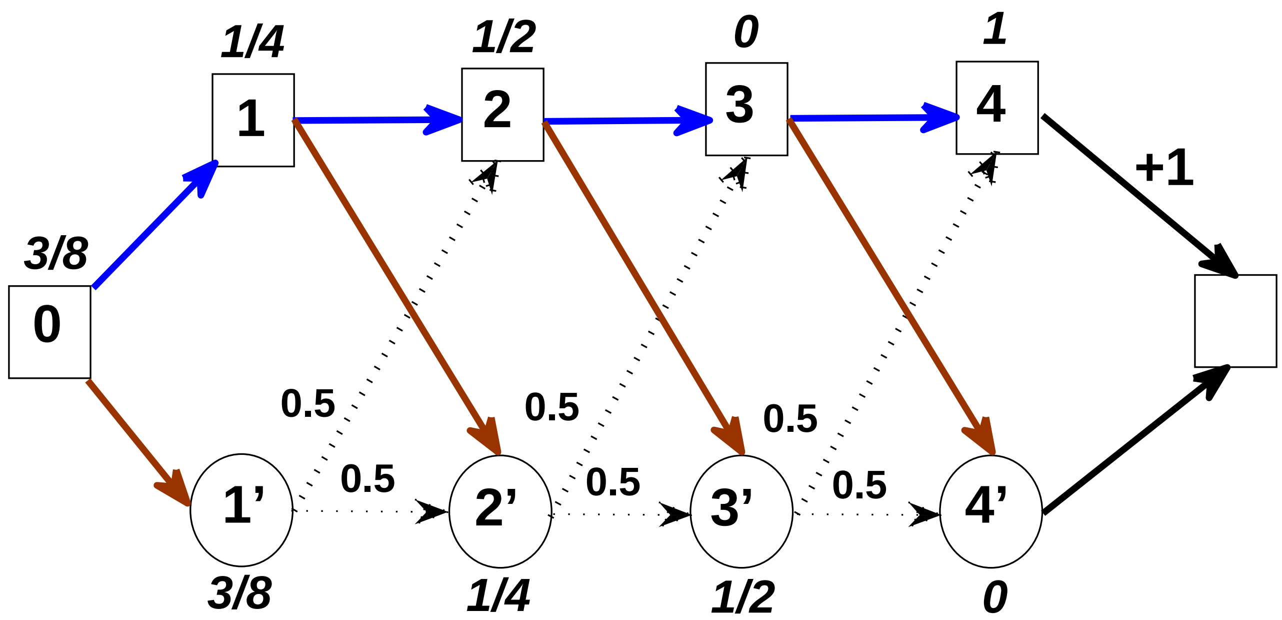
$$r_{ie} = \min_{j \in A_i}\{r_j\}, \ \forall i.$$

   If $r_e \geq 0 \ \forall i$, stop. The policy or BFS is optimal.

3. For every state i, if $r_{ie} < 0$, select x_{ie} be the entering basic variable to replace the current basic variable in state i; otherwise, keep the current basic variable in the basis.

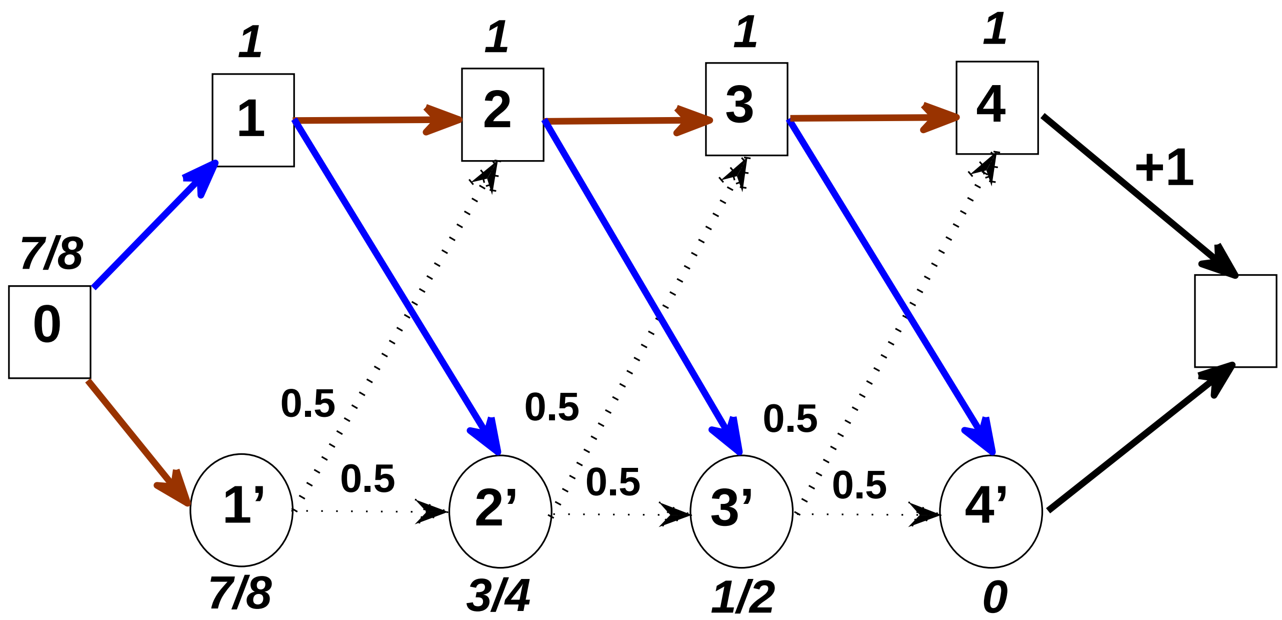
4. Update basis: update B and A_B and return to Step 1.

The (Classic) Policy Iteration

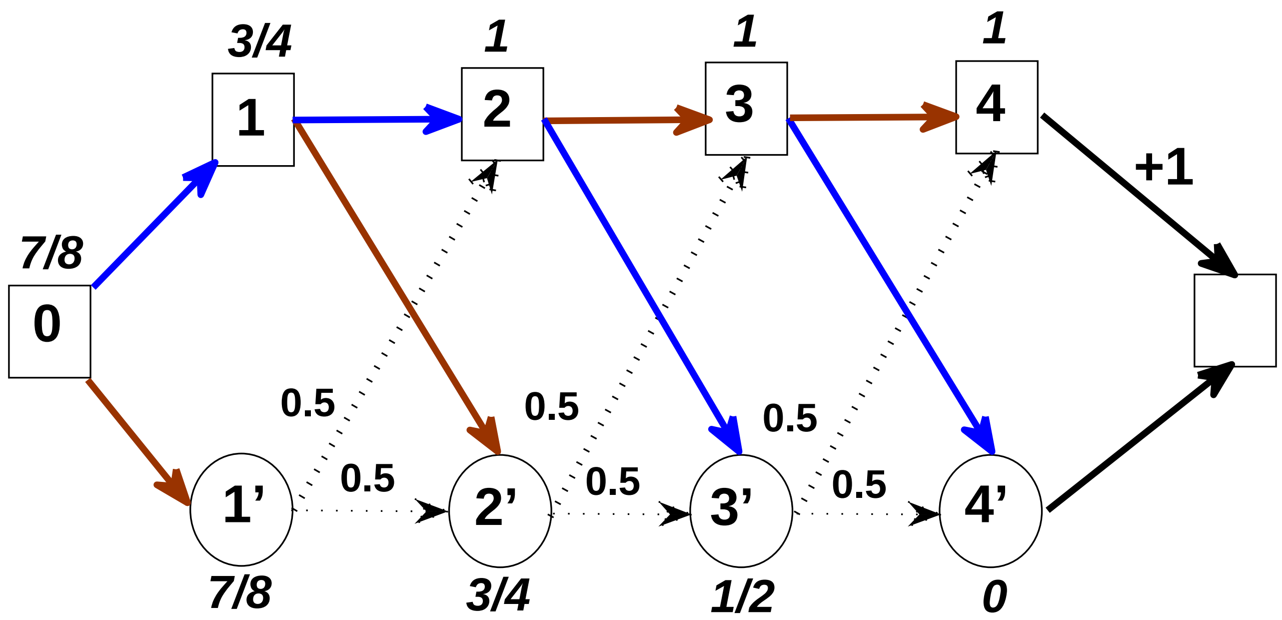


The Classic Policy Iteration (PI): simultaneously switch one action with the largest improvement rate in each state; new dual values on each state when actions in red are taken.

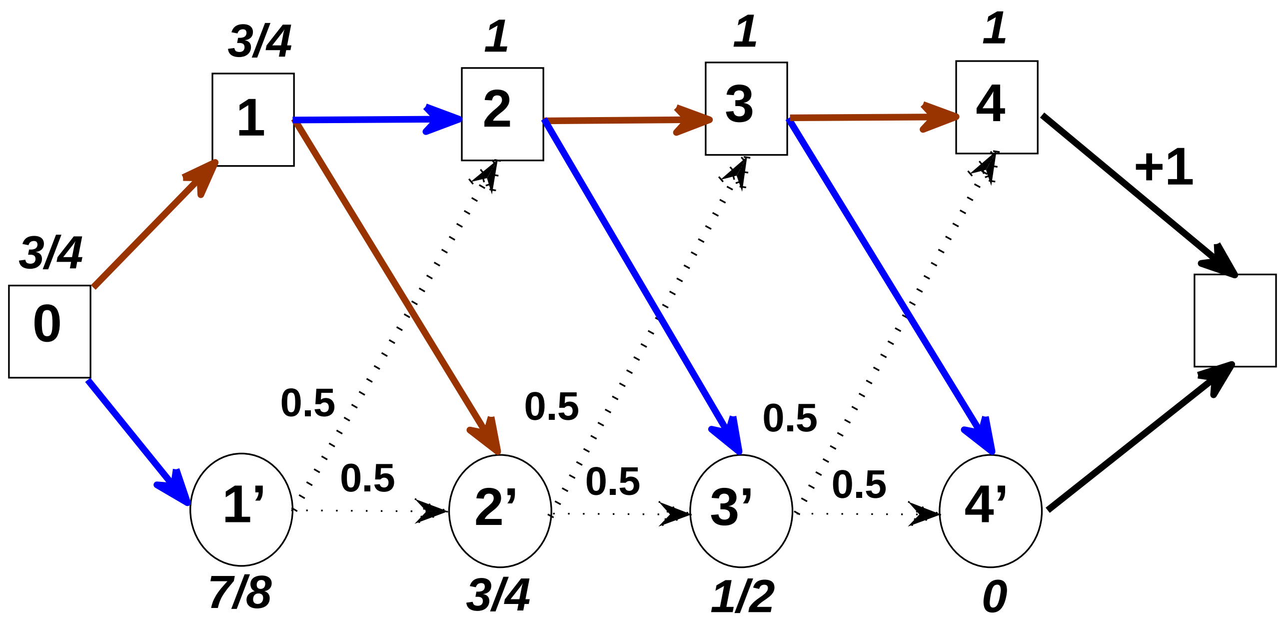
The Simplex or Simple Policy Iteration: index rule



The Simplex or Simple Policy Index-Rule Iteration: switch one action with the largest improvement rate in the lowest-indexed state; new dual values on each state when actions in red are taken.



The Simplex or Simple Policy Index-Rule Iteration II: New values on each state when actions in red are taken.



The Simplex or Simple Policy Index-Rule Iteration III: New values on each state when actions in red are taken.

Complexity of the Policy-Iteration and Simplex Methods

- In practice, the Policy Iteration (PI) method, including the simple policy iteration or Simplex method, has been remarkably successful and shown to be most effective and widely used.

- Mansour and Singh in 1994 gave an upper bound on the number of iterations, $2^m/m$, for the policy-iteration method when each state has 2 actions.

- A negative result, similar to Klee and Minty (1972), of Melekopoglou and Condon (1990) showed that a simple Policy Iteration method, where in each iteration only the action for the state with the smallest index is updated, needs an exponential number of iterations to compute an optimal policy for a specific MDP problem regardless of the discount rates.

- In the past 50 years, many efforts have been made to resolve the worst-case complexity issue of the Policy Iteration method or the Simplex method, and to answer the question: are they (strongly) polynomial-time algorithms?

The Discounted MDP Properties

Lemma 2 *The discounted MDP primal LP formulation has the following properties:*

1. *The feasible set is bounded. More precisely, for every feasible* $\mathbf{x} \geq \mathbf{0}$, $\mathbf{e}^T\mathbf{x} = \frac{m}{1-\gamma}$
2. *There is a one-to-one correspondence between a stationary policy of the original discounted MDP and a basic feasible solution (BFS) of the primal.*
3. *Every policy or BFS basis has the Leontief substitution form* $A_\pi = I - \gamma P_\pi$.
4. *Let* $\mathbf{x}^\pi$ *be a basic feasible solution. Then any basic variable, say* $\mathbf{x}_i^\pi$, *has its value* $1 \leq \mathbf{x}_i^\pi \leq \frac{m}{1-\gamma}$.

Technical Results of the Classic PI

Proposition 1 *For the PI iteration $k = 0, 1, ...,$*

- $\mathbf{c}^T\mathbf{x}^k - z^* = \mathbf{e}^T\mathbf{y}^k - \mathbf{e}^T\mathbf{y}^*$ *where $\mathbf{y}^*$ is the optimal dual solution.*
- $\mathbf{y}^k \geq \mathbf{y}^{k+1} \geq \mathbf{y}^*$.
- *For every state i,*

$$\mathbf{y}_i^k - \mathbf{y}_i^* \geq r_j^*, \; j \in B^k \cap A_i$$

  where B^k is the policy/basis set at step k and $\mathbf{r}^* = \mathbf{c} - A^T\mathbf{y}^*$.
- $\|\mathbf{y}^{k+1} - \mathbf{y}^*\|_\infty \leq \gamma\|\mathbf{y}^k - \mathbf{y}^*\|_\infty$.

Technical Results of the Classic PI

To prove $\mathbf{y}_i^k - \mathbf{y}_i^* \geq r_j^*,\ j \in B^k \cap A_i$:

$$\mathbf{r}_{B^k}^* = \mathbf{c}_{B^k} - A_{B^k}^T \mathbf{y}^* = A_{B^k}^T \mathbf{y}^k - A_{B^k}^T \mathbf{y}^* = A_{B^k}^T (\mathbf{y}^k - \mathbf{y}^*)$$

so that

$$\mathbf{y}^k - \mathbf{y}^* = (A_{B^k}^T)^{-1} \mathbf{r}_{B^k}^* \geq \mathbf{r}_{B^k}^*.$$

To prove $\|\mathbf{y}^{k+1} - \mathbf{y}^*\|_\infty \leq \gamma \|\mathbf{y}^k - \mathbf{y}^*\|_\infty$:

$$A_{B^{k+1}}^T (\mathbf{y}^{k+1} - \mathbf{y}^k) = \mathbf{c}_{B^{k+1}} - A_{B^{k+1}}^T \mathbf{y}^k = \mathbf{r}_{B^{k+1}}^k \leq \mathbf{c}_{B^*} - A_{B^*}^T \mathbf{y}^k = A_{B^*}^T \mathbf{y}^* - A_{B^*}^T \mathbf{y}^k.$$

Thus,

$$\begin{aligned}
\mathbf{y}^{k+1} - \mathbf{y}^k &\leq \mathbf{y}^{k+1} - \mathbf{y}^k - \gamma P_{B^{k+1}}^T (\mathbf{y}^{k+1} - \mathbf{y}^k) \\
&= A_{B^{k+1}}^T (\mathbf{y}^{k+1} - \mathbf{y}^k) \leq A_{B^*}^T \mathbf{y}^* - A_{B^*}^T \mathbf{y}^k \\
&= \mathbf{y}^* - \mathbf{y}^k + \gamma P_{B^*}^T (\mathbf{y}^k - \mathbf{y}^*)
\end{aligned}$$

which implies $\mathbf{0} \leq \mathbf{y}^{k+1} - \mathbf{y}^* \leq \gamma P_{B^*}^T (\mathbf{y}^k - \mathbf{y}^*)$ and the desired result.

Precise Complexity Results

- The classic simplex and policy iteration methods, with the greedy pivoting rule, are a strongly polynomial-time algorithm for MDP with fixed discount rate. The method terminates in a number of steps bounded by $\frac{mn}{1-\gamma} \cdot \log\left(\frac{m^2}{1-\gamma}\right)$, and each step uses at most $O(mn)$ arithmetic operations, where n is the total number of actions.

- The classic policy(strategy)-iteration method terminates in no more

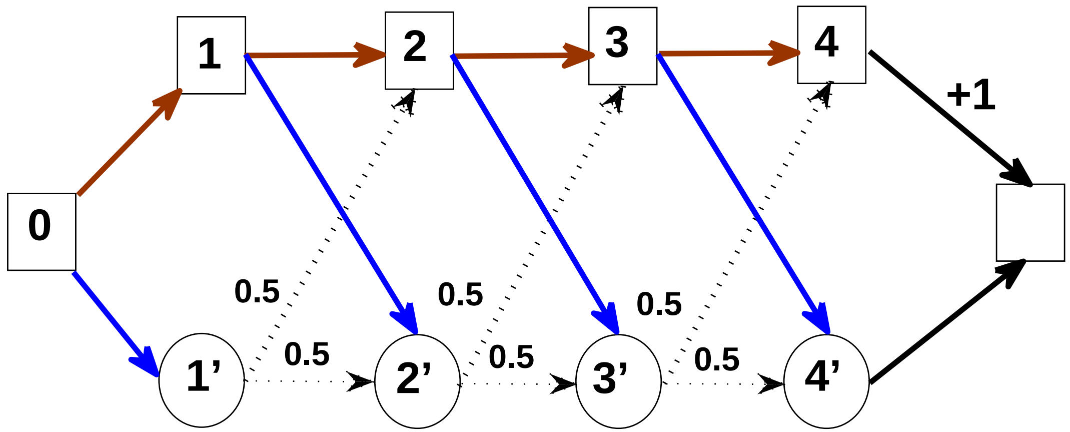
$$\frac{n}{1-\gamma} \cdot \log\left(\frac{m}{1-\gamma}\right),$$

steps and each step uses at most m^2n arithmetic operations (Hansen, Miltersen, and Zwick, September 2010).

The Shapley Two-Person Zero-Sum Stochastic Game

- Similar to the Markov decision process, but the states is partitioned to two sets where one is to maximize and the other is to minimize.
- It has no linear programming formulation, and it is unknown if it can be solved in polynomial time in general.
- For a fixed discount rate, it can be solved in polynomial time (Littman 1996) using the value iteration method.
- Hansen, Miltersen and Zwick (2010) very recently proved that the strategy iteration method solves it in strongly polynomial time when discount rate is fixed. This is the first strongly polynomial time algorithm for solving the discounted game.

A Markov Decision/Game Process Example



A Markov Game Process Example: states $\{3, 4\}$ want to maximize while states $\{0, 1, 2\}$ want to minimize.

The Fixed-Point Model of the 2BZSG

The equilibrium Cost-to-Go values for all states meet the Bellman Principle:

$$\begin{aligned} y_i &= \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\}, \forall i \in I^- \\ y_i &= \max_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\}, \forall i \in I^+ \end{aligned}$$

where $\mathcal{A}_i$ represents all actions available in state i, $\mathbf{p}_j$ is the state transition probabilities from state i to all states when action j in state i is taken.

The policy induced by the fixed point is called the Nash-Equilibrium policy.

There is no LP formulation of this fixed-point problem.

The Strategy Iteration Method for MDP

0. Initialize Start from any policy set I^- of the min-player.

1. For given I^-, find the maximal policy set I^+ of the max-player using the Policy-Iteration.

2. Denote the joint policy as B, and Compute $\mathbf{x}_B = (A_B)^{-1}\mathbf{e} \geq \mathbf{0}$, $\mathbf{y}^T = \mathbf{c}_B^T(A_B)^{-1}$, and $\mathbf{r} = \mathbf{c} - A^T\mathbf{y}$. Note that $r_j \leq 0, \forall j \in \mathcal{A}_i$ and $i \in I^+$.

3. Select

$$r_{ie} = \min_{j \in A_i}\{r_j\}, \ \forall i \in I^-.$$

   If $r_{ie} \geq 0 \ \forall i$, stop. The policy or BFS is the Nach-Equilibrium.

4. For every state $i \in I^-$, if $r_{ie} < 0$, select x_{ie} be the entering basic variable to replace the current basic variable in state $i \in I^-$; otherwise, keep the current basic variable in the basis.

5. Update basis: update B^- and return to Step 1.

The Value-Iteration Method for MDP/Game

$$y_i = \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\}, \forall i \in I^-$$
$$y_i = \max_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}\}, \forall i \in I^+$$

Value Iteration Method: Starting with any vector $\mathbf{y}^0$, then iteratively update it

$$y_i^{k+1} = \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^k\}, \ \forall i \in I^-$$

and

$$y_i^{k+1} = \max_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^k\}, \ \forall i \in I^+.$$

Convergence of VI

Proposition 2 *The following results hold.*

- *Let* $\mathbf{y}^*$ *be the fixed-point. Then*

$$\|\mathbf{y}^{k+1} - \mathbf{y}^*\|_\infty \leq \gamma \|\mathbf{y}^k - \mathbf{y}^*\|_\infty, \ \forall k.$$

- *For MDP, let* $\mathbf{y}^* \leq \mathbf{y}^0$ *and* $\mathbf{y}^1 \leq \mathbf{y}^0$. *Then*

$$\mathbf{y}^* \leq \mathbf{y}^{k+1} \leq \mathbf{y}^k, \ \forall k.$$

VI Variants: The Randomized Value-Iteration Method for MDP

Rather than go through all state values in each iteration, we modify the VI method, call it RamdomVI: In the kth iteration, randomly select a subset of states S^k and do

$$y_i^{k+1} = \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^k\}, \ \forall i \in S^k.$$

In RandomVI, we only update a subset of state values at random in each iteration.

VI Variants: The Cyclic Value-Iteration Method for MDP

Here is another modification, called CyclicVI: In the kth iteration do

- Initialize $\tilde{\mathbf{y}}^k = \mathbf{y}^k$.
- For $i = 1$ to m

$$\tilde{y}_i^k = \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \tilde{\mathbf{y}}^k\}$$

- $\mathbf{y}^{k+1} = \tilde{\mathbf{y}}^k$.

In the CyclicVI method, as soon as a state value is updated, we use it to update the rest of state values.

VI Variants: Randomly Permuted CyclicVI Method for MDP

In the CyclicVI method, rather than with the fixed cycle order from 1 to m, we follow a random permutation order, or sample without replacement to update the state values. More precisely, in the kth iteration do

0. Initialize $\tilde{\mathbf{y}}^k = \mathbf{y}^k$ and $S^k = \{1, 2, ..., m\}$
1. – Randomly select $i \in S^k$
   –
   $$\tilde{y}_i^k = \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \tilde{\mathbf{y}}^k\}$$
   – If $S^k \neq \emptyset$, remove i from S^k and return to Step 1.
3. $\mathbf{y}^{k+1} = \tilde{\mathbf{y}}^k$.

We call it the randomly permuted CyclicVI or RPCyclicVI in short.

VI Variants: VI Method for MDP Based on Samples

Recall VI for MDP: Starting with any vector $\mathbf{y}^0$, then iteratively update it

$$y_i^{k+1} = \min_{j \in \mathcal{A}_i} \{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^k\}, \ \forall i.$$

But $\mathbf{p}_j$ is not exactly known but samples can be drawn to find an empirical distribution $\tilde{\mathbf{p}}_j$':

Let N_j be the total number of samples when action j being used

$$\tilde{\mathbf{p}}_{ij} = \frac{\#\text{ of samples ending in state } i}{N_j}.$$

How many samples are needed to find an (approximate) optimal policy: a policy whose objective value is less than the minimal one plus a given ϵ?

VI Variants: Online State-Aggregation

Online state-aggregation: during the process of VI, aggregate the states into a single cluster if their cost-to-go values are close.

Then, in the VI, sample one state per cluster to update cost-to-go values and assume all states in the same cluster have a "same" cost-to-go value.

How to correct possible errors of some sates in a wrong cluster? How to analyze the sample and computational complexities comparing to the original VI method?

Remarks and Open Questions I

- The performance of the simplex method is very sensitive to the pivoting rule.
- Tatonnement and decentralized process works under the Markov property.
- Greedy or Steepest Descent works when there is a discount!
- Multi-updates or pivots work better than a single-update does; policy iteration vs. simplex.
- The proof techniques are generalized to solving general linear programs by Kitahara and Mizuno (2010).

Remarks and Open Questions II

- Can the iteration bound for the Simplex method be reduced to linear in the number of actions?
- Is the Simplex or Policy iteration method polynomial for the MDP regardless of discount rate γ or input data? (It has been proved to be true for the Simplex method on the deterministic MDP, Post & Y 2015.)
- Is there an MDP algorithm whose running time is strongly polynomial regardless of discount rate γ and other input data?
- Is there a Stochastic Game algorithm whose running time is polynomial regardless of discount rate γ? Even for deterministic game?
- Is there a strongly polynomial-time algorithm for LP?
- Development of approximate policy and/or value iteration methods to accelerate the solution speed in practice.