Stochastic Dynamic Programming

Fatih Cavdur

fatihcavdur@uludag.edu.tr

Example

Example: For a price of $1/gallon, the Safeco Supermarket chain has purchased 6 gallons of milk from a local dairy. Each gallon of milk is sold in the chain’s three stores for $2/gallon. The dairy must buy back for 50¢/gallon any milk that is left at the end of the day. Unfortunately for Safeco, demand for each of the chain’s three stores is uncertain. Past data indicate that the daily demand at each store is as shown in the below table. Safeco wants to allocate the 6 gallons of milk to the three stores so as to maximize the expected net daily profit (revenues less costs) earned from milk. Use dynamic programming to determine how Safeco should allocate the 6 gallons of milk among the three stores.

Example

	Daily Demand	Probability
Store 1	1	.6
	2	.0
	3	.4
Store 2	1	.5
	2	.1
	3	.4
Store 3	1	.4
	2	.3
	3	.3

Table: Problem Data

Example

$r_t(g_t)$ = expected revenue earned from g_t gallons assigned to store t

$f_t(x)$ = maximum expected revenue earned from x gallons assigned to stores $t, t+1, \dots, 3$

We have,

$$f_3(x) = r_3(x)$$

$$f_t(x) = \max_{g_t}\{r_t(g_t) + f_{t+1}(x - g_t)\}, \quad t = 1,2$$

$$r_3(0) = \$0 \quad r_3(1) = \$2.00 \quad r_3(2) = \$3.40 \quad r_3(3) = \$4.35$$

$$r_2(0) = \$0 \quad r_2(1) = \$2.00 \quad r_2(2) = \$3.25 \quad r_2(3) = \$4.35$$

$$r_1(0) = \$0 \quad r_1(1) = \$2.00 \quad r_1(2) = \$3.10 \quad r_1(3) = \$4.20$$

Example

Stage 3 Computations:

$$f_3(0) = r_3(0) = 0.00 \Rightarrow g_3(0) = 0$$

$$f_3(1) = r_3(1) = 2.00 \Rightarrow g_3(1) = 1$$

$$f_3(2) = r_3(2) = 3.40 \Rightarrow g_3(2) = 2$$

$$f_3(3) = r_3(3) = 4.35 \Rightarrow g_3(3) = 3$$

Example

$$f_2(0) = r_2(0) + f_3(0-0) = 0.00 \Rightarrow g_2(0) = 0$$

$$f_2(1) = \max\begin{Bmatrix} r_2(0) + f_3(1-0) = 2.00 \\ r_2(1) + f_3(1-1) = 2.00 \end{Bmatrix} \Rightarrow g_2(1) = 0 \vee 1$$

$$f_2(2) = \max\begin{Bmatrix} r_2(0) + f_3(2-0) = 0.00 + 3.40 = 3.40 \\ r_2(1) + f_3(2-1) = 2.00 + 2.00 = 4.00 \\ r_2(2) + f_3(2-2) = 3.25 + 0.00 = 3.25 \end{Bmatrix} \Rightarrow g_2(2) = 1$$

$$f_2(3) = \max\begin{Bmatrix} r_2(0) + f_3(3-0) = 0.00 + 4.35 = 4.35 \\ r_2(1) + f_3(3-1) = 2.00 + 3.40 = 5.40 \\ r_2(2) + f_3(3-2) = 3.25 + 2.00 = 5.25 \\ r_2(3) + f_3(3-3) = 4.35 + 0.00 = 4.35 \end{Bmatrix} \Rightarrow g_2(3) = 1$$

Example

$$f_2(4) = \max \begin{Bmatrix} r_2(1) + f_3(4-1) = 2.00 + 4.35 = 6.35 \\ r_2(2) + f_3(4-2) = 3.25 + 3.40 = 6.65 \\ r_2(3) + f_3(4-3) = 4.35 + 2.00 = 6.35 \end{Bmatrix} \Rightarrow g_2(4) = 2$$

$$f_2(5) = \max \begin{Bmatrix} r_2(2) + f_3(5-2) = 3.25 + 4.35 = 7.60 \\ r_2(3) + f_3(5-3) = 4.35 + 3.40 = 7.75 \end{Bmatrix} \Rightarrow g_2(5) = 3$$

$$f_2(6) = r_2(3) + f_3(6-3) = 4.35 + 4.35 = 8.70 \Rightarrow g_2(6) = 3$$

Example

Stage 1 Computations:

$$f_1(6) = \max \begin{Bmatrix} r_1(0) + f_2(6-0) = 0.0 + 8.70 = 8.70 \\ r_1(1) + f_2(6-1) = 2.0 + 7.75 = 9.75 \\ r_1(2) + f_2(6-2) = 3.1 + 6.65 = 9.75 \\ r_1(3) + f_2(6-3) = 4.2 + 5.40 = 9.60 \end{Bmatrix} \Rightarrow g_1(6) = 1 \vee 2$$

A Stochastic Inventory Model

Consider the following three-period inventory problem. At the beginning of each period, a firm must determine how many units should be produced during the current period. During a period in which x units are produced, a production cost $c(x)$ is incurred, where $c(0) = 0$, and for $x > 0$, $c(x) = 3 + 2x$. Production during each period is limited to at most 4 units. After production occurs, the period's random demand is observed. Each period's demand is equally likely to be 1 or 2 units. After meeting the current period's demand out of current production and inventory, the firm's end-of-period inventory is evaluated, and a holding cost of $1 per unit is assessed. Because of limited capacity, the inventory at the end of each period cannot exceed 3 units. It is required that all demand be met on time. Any inventory on hand at the end of period 3 can be sold at $2 per unit. At the beginning of period 1, the firm has 1 unit of inventory. Use dynamic programming to determine a production policy that minimizes the expected net cost incurred during the three periods.

A Stochastic Inventory Model

We have, for $t = 3$,

$$f_3(i) = \min_x \left\{ c(x) + \frac{(i + x - 1)}{2} + \frac{(i + x - 2)}{2} - \frac{(2)(i + x - 1)}{2} - \frac{(2)(i + x - 2)}{2} \right\}$$

and, for $t = 2,1$,

$$f_t(i) = \min_x \left\{ c(x) + \frac{(i + x - 1)}{2} + \frac{(i + x - 2)}{2} + \frac{f_{t+1}(i + x - 1)}{2} + \frac{f_{t+1}(i + x - 2)}{2} \right\}$$

A Stochastic Inventory Model

Stage 3 Computations:

$$f_3(i) = \min_x \left\{ c(x) + \frac{(i+x-1)}{2} + \frac{(i+x-2)}{2} - \frac{(2)(i+x-1)}{2} - \frac{(2)(i+x-2)}{2} \right\}$$

i	x	$c(x)$	Expected Holding Cost $i + x - \frac{3}{2}$	Expected Salvage Value $2i + 2x - 3$	Expected Total Cost	$f_3(i); x_3(i)$
3	0	0	3/2	3	-3/2	$f_3(3) = -3/2; x_3(0) = 0$
3	1	5	5/2	5	5/2	
2	0	0	1/2	1	-1/2	$f_3(2) = -1/2; x_3(0) = 0$
2	1	5	3/2	3	7/2	
2	2	7	5/2	5	9/2	
1	1	5	1/2	1	9/2	$f_3(1) = 9/2; x_3(0) = 1$
1	2	7	3/2	3	11/2	
1	3	9	5/2	5	13/2	
0	2	7	1/2	1	13/2	$f_3(0) = 13/2; x_3(0) = 2$
0	3	9	3/2	3	15/2	
0	4	11	5/2	5	17/2	

A Stochastic Inventory Model

Stage 2 Computations:

$$f_2(i) = \min_x \left\{ c(x) + \frac{(i+x-1)}{2} + \frac{(i+x-2)}{2} + \frac{f_3(i+x-1)}{2} + \frac{f_3(i+x-2)}{2} \right\}$$

A Stochastic Inventory Model

i	x	$c(x)$	Expected Holding Cost $i+x-\frac{3}{2}$	Expected Future Cost $\frac{f_3(i+x-1)}{2}+\frac{f_3(i+x-2)}{2}$	Expected Total Cost	$f_2(i); x_2(i)$
3	0	0	3/2	2	7/2	$f_2(3)=\frac{7}{2}$ $x_2(3)=0$
3	1	5	5/2	-1	13/2	
2	0	0	1/2	11/2	6	$f_2(2)=6$ $x_2(2)=0$
2	1	5	3/2	2	17/2	
2	2	7	5/2	-1	17/2	
…	…	…	…	…	…	…
0	2	7	1/2	11/2	13	
0	3	9	3/2	2	25/2	$f_2(0)=\frac{25}{2}$ $x_2(0)=3$
0	4	11	5/2	-1	25/2	$f_2(0)=\frac{25}{2}$ $x_2(0)=4$

A Stochastic Inventory Model

Stage 1 Computations:

$$f_1(i) = \min_x \left\{ c(x) + \frac{(i+x-1)}{2} + \frac{(i+x-2)}{2} + \frac{f_2(i+x-1)}{2} + \frac{f_2(i+x-2)}{2} \right\}$$

i	x	$c(x)$	Expected Holding Cost $i+x-\frac{3}{2}$	Expected Future Cost $\frac{f_2(i+x-1)}{2}+\frac{f_2(i+x-2)}{2}$	Expected Total Cost	$f_1(i); x_1(i)$
1	1	7	1/2	23/2	17	
1	2	9	3/2	33/4	67/4	
1	3	11	5/2	19/4	65/4	$f_1(1) = \frac{65}{4}$ $x_1(1) = 3$

Gambler’s Problem

A gambler has \$2. She is allowed to play a game of chance four times, and her goal is to maximize her probability of ending up with a least \$6. If the gambler bets b dollars on a play of the game, then with probability .40, she wins the game and increases her capital position by b dollars; with probability .60, she loses the game and decreases her capital by b dollars. On any play of the game, the gambler may not bet more money than she has available. Determine a betting strategy that will maximize the gambler’s probability of attaining a wealth of at least \$6 by the end of the fourth game. We assume that bets of zero dollars (that is, not betting) are permissible.

Gambler’s Problem

We let $f_t(d)$ be the probability that the gambler will have at least \$6 by the end of given that she has d dollars immediately before the game is played for the tth time and given that she acts optimally.

If the gambler playing the game for the 4th and final time, her optimal strategy is clear:

- If she has \$6 or more, don’t bet anything.
- If she has less than \$6, bet enough money to ensure (if possible) that she will have \$6 if she wins the last game.

Gambler's Problem

We hence have the following:

$$
\begin{aligned}
&f_4(0) = .0 \Rightarrow b_4(0) = \$0 \\
&f_4(1) = .0 \Rightarrow b_4(1) = \$0, \$1 \\
&f_4(2) = .0 \Rightarrow b_4(2) = \$0, \$1, \$2 \\
&f_4(3) = .4 \Rightarrow b_4(3) = \$3 \\
&f_4(4) = .4 \Rightarrow b_4(4) = \$2, \$3, \$4 \\
&f_4(5) = .4 \Rightarrow b_4(5) = \$1, \$2, \$3, \$4, \$5
\end{aligned}
$$

For $d \geq 6$,

$$f_4(d) = 1 \Rightarrow b_4(d) = \$0, \$1, \dots, \$(d-6)$$

We can write,

$$f_t(d) = \max_{b \in \{0, \dots d\}} \{.\,4 f_{t+1}(d+b) + .6 f_{t+1}(d-b)\}$$

Gambler’s Problem

Stage 3 Computations:

$$f_3(0) = 0 \Rightarrow b_3(0) = 0$$

$$f_3(1) = \max_b \begin{Bmatrix} .4f_4(1) + .6f_4(1) = 0 \\ .4f_4(2) + .6f_4(0) = 0 \end{Bmatrix} \Rightarrow b_3(1) = 0 \vee 1$$

$$f_3(2) = \max_b \begin{Bmatrix} .4f_4(2) + .6f_4(2) = .00 \\ .4f_4(3) + .6f_4(1) = .16 \\ .4f_4(4) + .6f_4(0) = .16 \end{Bmatrix} \Rightarrow b_3(2) = 1 \vee 2$$

If we continue similarly,

$$f_3(5) = \max_b \begin{Bmatrix} .4f_4(5) + .6f_4(5) = .40 \\ .4f_4(6) + .6f_4(4) = .64 \\ .4f_4(7) + .6f_4(3) = .64 \\ .4f_4(8) + .6f_4(2) = .40 \\ .4f_4(1) + .6f_4(9) = .40 \\ .4f_4(10) + .6f_4(0) = .40 \end{Bmatrix} \Rightarrow b_3(5) = 1 \vee 2$$

Gambler’s Problem

Stage 2 Computations:

$$f_2(0) = 0 \Rightarrow b_2(0) = 0$$

$$f_2(1) = \max_b \begin{Bmatrix} .4f_3(1) + .6f_3(1) = .000 \\ .4f_3(2) + .6f_3(0) = .064 \end{Bmatrix} \Rightarrow b_2(1) = 1$$

$$f_2(2) = \max_b \begin{Bmatrix} .4f_3(2) + .6f_3(2) = .16 \\ .4f_3(3) + .6f_3(1) = .16 \\ .4f_3(4) + .6f_3(0) = .16 \end{Bmatrix} \Rightarrow b_2(2) = 1 \vee 2 \vee 3$$

If we continue similarly,

$$f_2(5) = \max_b \begin{Bmatrix} .4f_3(5) + .6f_3(5) = .640 \\ .4f_3(6) + .6f_3(4) = .640 \\ .4f_3(7) + .6f_3(3) = .640 \\ .4f_3(8) + .6f_3(2) = .496 \\ .4f_3(1) + .6f_3(9) = .400 \\ .4f_3(10) + .6f_3(0) = .400 \end{Bmatrix} \Rightarrow b_3(5) = 0 \vee 1 \vee 2$$

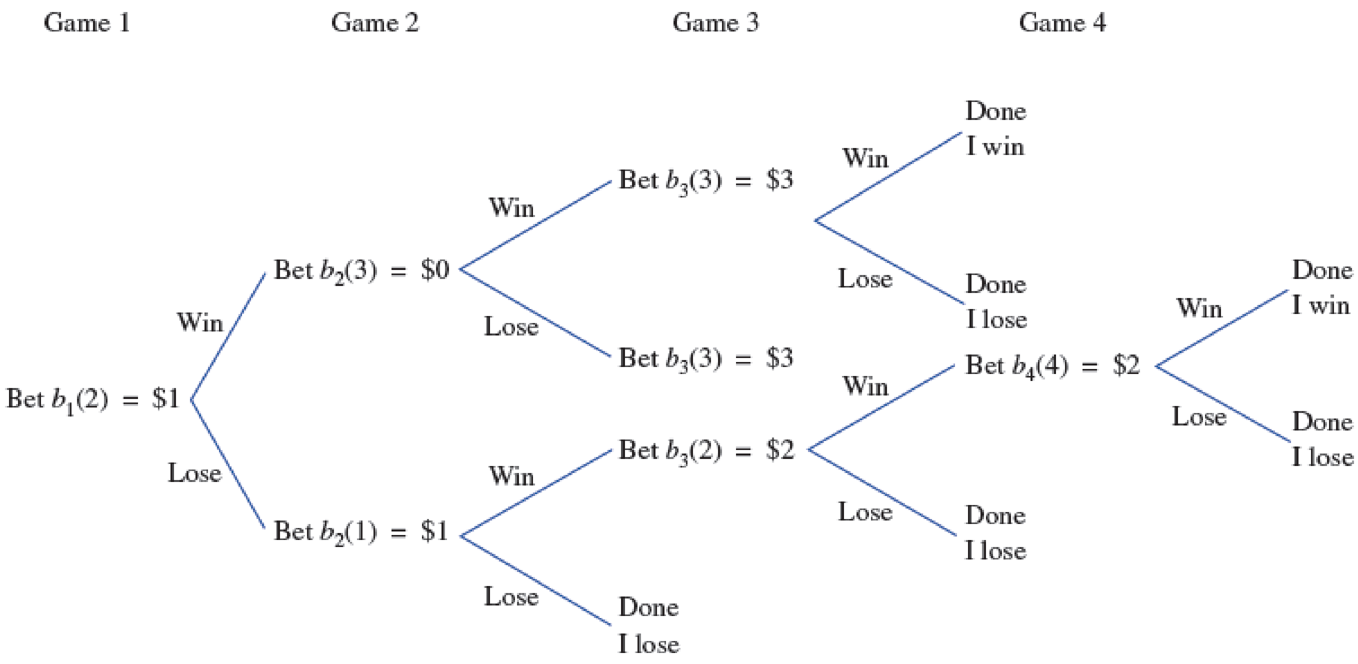
Gambler's Problem

Stage 1 Computations:

$$f_1(2) = \max_b \begin{Bmatrix} .4f_2(2) + .6f_2(2) = .1600 \\ .4f_2(3) + .6f_2(1) = .1984 \\ .4f_2(4) + .6f_2(0) = .1984 \end{Bmatrix} \Rightarrow b_2(2) = 1 \vee 2$$

Hence, the gambler has a chance of .1984 reaching $6.

Gambler’s Problem



Tennis Player

A tennis player has two types of serves, a hard (H) and a soft (S) one. The probability that her hard serve will land in bounds is p_H and the probability that her soft serve will land in bounds is p_S. If her hard serve lands in bounds, there is a probability w_H that she will win the point. If her soft serve lands in bounds, there is a probability w_s that she will win the point. We assume that $p_H < p_S$ and $w_H > w_S$. Her goal is to maximize the probability of winning the point. Use DP to help her! ☺

Tennis Player

We let f_t be the probability that she wins the point if she is about to take her tth service, $t = 1,2$.

$$f_2 = \max_x \begin{Bmatrix} p_H w_H \\ p_S w_S \end{Bmatrix} = p_S w_S \Rightarrow x_2 = S$$

$$f_1 = \max_x \begin{Bmatrix} p_H w_H + (1 - p_H) f_2 \\ p_S w_S + (1 - p_S) f_2 \end{Bmatrix}$$

Tennis Player

If we assume, $p_S w_S > p_H w_H$,

$$\Rightarrow p_H w_H + (1 - p_H) f_2 \geq p_S w_S + (1 - p_S) f_2$$
$$\Rightarrow p_H w_H + (1 - p_H) p_S w_S \geq p_S w_S + (1 - p_S) p_S w_S$$
$$\Rightarrow p_H w_H \geq p_S w_S + (1 + p_H - p_S)$$

We now assume, $p_S w_S \leq p_H w_H$, where she should serve hard on both. We then have

$$f_2 = \max_x \begin{Bmatrix} p_H w_H \\ p_S w_S \end{Bmatrix} = p_H w_H \Rightarrow x_2 = H$$

She should serve hard on the first if

$$\Rightarrow p_H w_H + (1 - p_H) f_2 \geq p_S w_S + (1 - p_S) f_2$$
$$\Rightarrow p_H w_H + (1 - p_H) p_H w_H \geq p_S w_S + (1 - p_S) p_H w_H$$
$$\Rightarrow p_S w_S \leq p_H w_H + (1 + p_S - p_H)$$

Markov Decision Processes (MDP)

An MDP is described by the following information:

- State Space
- Decision Set
- Transition Probabilities
- Expected Rewards

Markov Decision Processes (MDP)

At the beginning of each week, a machine is in one of four conditions (states): excellent (E), good (G), average (A), or bad (B). The weekly revenue earned by a machine in each type of condition is as follows: excellent, \$100; good, \$80; average, \$50; bad, \$10. After observing the condition of a machine at the beginning of the week, we have the option of instantaneously replacing it with an excellent machine, which costs \$200. The quality of a machine deteriorates over time, as given in the following matrix. For this situation, determine the state space, decision sets, transition probabilities, and expected rewards.

Markov Decision Processes (MDP)

$$\mathbf{P} = \begin{array}{c} \\ E \\ G \\ A \\ B \end{array} \begin{array}{cccc} E & G & A & B \\ \end{array}$$

$$\mathbf{P} = \begin{array}{c|cccc} & E & G & A & B \\ \hline E & .7 & .3 & .0 & 0.0 \\ G & .0 & .7 & .3 & 0.0 \\ A & .0 & .0 & .6 & 0.4 \\ B & .0 & .0 & .0 & 1.0 \end{array}$$

State Set: $S = \{E, G, A, B\}$

Decision Set: $R = \{R, K\}$, replace and do not replace (keep)

We have

$D(E) = \{K\}$ and $D(G) = D(A) = D(B) = \{R, K\}$

We are given the following transition probabilities:

$$\begin{array}{cccc} p(E|E,K) = .7 & p(G|E,K) = .3 & \dots & p(B|E,K) = 0.0 \\ \vdots & \vdots & \ddots & \vdots \\ p(E|B,K) = .0 & p(G|B,K) = .0 & \dots & p(B|B,K) = 1.0 \end{array}$$

Markov Decision Processes (MDP)

If we replace a machine with an excellent machine, the transition probabilities will be the same as if we had begun the week with an excellent machine.

$$p(E|G,R) = p(E|A,R) = p(E|B,R) = .7$$

$$p(G|G,R) = p(G|A,R) = p(G|B,R) = .3$$

$$p(A|G,R) = p(A|A,R) = p(A|B,R) = .0$$

$$p(B|G,R) = p(B|A,R) = p(B|B,R) = .0$$

Markov Decision Processes (MDP)

If the machine is not replaced, then, during the week, we receive the revenues given in the problem.

$$r_{E,K} = \$100, r_{G,K} = \$80, r_{A,K} = \$50, r_{B,K} = \$10$$

If we replace the machine,

$$r_{E,R} = r_{G,R} = r_{A,R} = r_{B,R} = -\$100$$

Markov Decision Processes (MDP)

Definition:

A policy is a rule that specifies how each period’s decision is chosen.

Definition:

A policy δ is a stationary policy if whenever the state i, the policy δ chooses (independently of the period) the same decision $\delta(i)$.

Markov Decision Processes (MDP)

δ: an arbitrary policy

Δ: the set of all policies

X_t: random variable for the state of MDP at the beginning of period t

X_1: given state of the process at the beginning of period 1 (initial state)

d_t: decision chosen during period t

$V_\delta(i)$: expected discounted reward earned during an infinite number of periods, given that at the beginning of period 1, state is i and stationary policy δ is followed.

Markov Decision Processes (MDP)

We can then write,

$$V_\delta(i) = E_\delta\left(\sum_{t=1}^{\infty} \beta^{t-1} r_{x_t d_t} | X_1 = i\right)$$

In a max problem,

$$V(i) = \max_{\delta \in \Delta} V_\delta(i)$$

In a min problem,

$$V(i) = \min_{\delta \in \Delta} V_\delta(i)$$

Markov Decision Processes (MDP)

Definition:

If a policy δ^* has the property that for all $i \in S$

$$V(i) = V_{\delta^*}(i)$$

then, δ^* is an optimal policy.

Markov Decision Processes (MDP)

We can use the following approaches to find the optimal stationary policy:

- Policy Iteration
- Linear Programming

- Value Iteration or Successive Approximations

Policy Iteration

$V_\delta(i)$ can be found by solving the following system of linear equations:

$$V_\delta(i) = r_{i,\delta(i)} + \beta \sum_{j=1}^{N} p\big(j|i, \delta(i)\big) V_\delta(j), \quad i = 1, \dots, N$$

Consider the following stationary policy in the machine replacement example:

$$\delta(E) = \delta(G) = K; \delta(A) = \delta(B) = R$$

We then have,

$$\begin{aligned} V_\delta(E) &= 100 + .9[.7V_\delta(E) + .3V_\delta(G)] \\ V_\delta(G) &= 80 + .9[.7V_\delta(G) + .3V_\delta(A)] \\ V_\delta(A) &= -100 + .9[.7V_\delta(E) + .3V_\delta(G)] \\ V_\delta(B) &= -100 + .9[.7V_\delta(E) + .3V_\delta(G)] \end{aligned}$$

By solving these,

$V_\delta(E) = 687.81$, $V_\delta(G) = 573.19$, $V_\delta(A) = 487.81$, and $V_\delta(B) = 487.81$

Howard's Policy Iteration

Step 1) Policy Evaluation: Choose a stationary policy δ and use the equations to find $V_\delta(i)$, $i = 1, \dots, N$

Step 2) Policy Improvement: For all states $i = 1, \dots, N$, compute

$$T_\delta(i) = \max_{d \in D(i)} \left\{ r_{i,d} + \beta \sum_{j=1}^{N} p(j|i,d) V_\delta(j) \right\}$$

If $T_\delta(i) = V_\delta(i), \forall i \Rightarrow \delta$ is an optimal policy.

If $T_\delta(i) > V_\delta(i), \exists i \Rightarrow \delta$ is not an optimal policy.

Modify δ to obtain δ' for which $V_{\delta'}(i) \geq V_\delta(i), \forall i$.

Return to Step (1) with policy δ'.

Howard’s Policy Iteration

Machine Replacement Example:

Consider the following stationary policy:

$\delta(E) = \delta(G) = K$ and $\delta(A) = \delta(B) = R$

We have found that

$V_\delta(E) = 687.81, V_\delta(G) = 573.19, V_\delta(A) = 487.81$, and $V_\delta(B) = 487.81$

Howard’s Policy Iteration

Now,

$$T_\delta(E) = V_\delta(E) = 687.81$$

$$\begin{aligned} T_\delta(G) &= \max\left\{\begin{matrix} -100 + .9[.7V_\delta(E) + .3V_\delta(G)] = 487.81 \\ 80 + .9[.7V_\delta(G) + .3V_\delta(A)] = V_\delta(G) = 572.19 \end{matrix}\right\} \\ &= 572.19 \Rightarrow \delta(G) = K \end{aligned}$$

$$\begin{aligned} T_\delta(A) &= \max\left\{\begin{matrix} -100 + .9[.7V_\delta(E) + .3V_\delta(G)] = 487.81 \\ 50 + .9[.6V_\delta(A) + .4V_\delta(B)] = 489.03 \end{matrix}\right\} \\ &= 489.03 \Rightarrow \delta(A) = K \end{aligned}$$

$$\begin{aligned} T_\delta(B) &= \max\left\{\begin{matrix} -100 + .9[.7V_\delta(E) + .3V_\delta(G)] = V_\delta(B) = 487.81 \\ 10 + .9V_\delta(B) = 449.03 \end{matrix}\right\} \\ &= 487.81 \Rightarrow \delta(G) = R \end{aligned}$$

Howard’s Policy Iteration

Since $T_\delta(i) > V_\delta(i)$, for $i = A$, the policy δ is not optimal. So replace it with δ' given as

$\delta'(E) = \delta'(G) = \delta'(A) = K$ and $\delta'(B) = R$

We now return to Step (1) and compute $V_{\delta'}(E) = 690.23$, $V_{\delta'}(G) = 575.50$, $V_{\delta'}(A) = 492.35$, and $V_{\delta'}(B) = 490.23$ by solving the following system. Note that $V_{\delta'}(i) > V_\delta(i), \forall i$.

$$\begin{aligned}
V_{\delta'}(E) &= 100 + .9[.7V_{\delta'}(E) + .3V_{\delta'}(G)] \\
V_{\delta'}(G) &= 80 + .9[.7V_{\delta'}(G) + .3V_{\delta'}(A)] \\
V_{\delta'}(A) &= 50 + .9[.6V_{\delta'}(A) + .4V_{\delta'}(G)] \\
V_{\delta'}(B) &= -100 + .9[.7V_{\delta'}(E) + .3V_{\delta'}(G)]
\end{aligned}$$

Howard’s Policy Iteration

Now apply the policy iteration procedure as follows:

$$T_{\delta'}(E) = V_{\delta'}(E) = 690.23$$

$$T_{\delta'}(G) = \max\left\{\begin{array}{c} -100 + .9[.7V_{\delta'}(E) + .3V_{\delta'}(G)] = 490.23 \\ 80 + .9[.7V_{\delta'}(G) + .3V_{\delta'}(A)] = V_{\delta'}(G) = 575.50 \end{array}\right\}$$
$$= 575.50 \Rightarrow \delta'(G) = K$$

$$T_{\delta'}(A) = \max\left\{\begin{array}{c} -100 + .9[.7V_{\delta'}(E) + .3V_{\delta'}(G)] = 490.23 \\ 50 + .9[.6V_{\delta}(A) + .4V_{\delta}(B)] = 492.35 \end{array}\right\}$$
$$= 492.35 \Rightarrow \delta(A) = K$$

$$T_{\delta'}(B) = \max\left\{\begin{array}{c} -100 + .9[.7V_{\delta'}(E) + .3V_{\delta'}(G)] = V_{\delta'}(B) = 490.23 \\ 10 + .9V_{\delta'}(B) = 451.21 \end{array}\right\}$$
$$= 490.23 \Rightarrow \delta(G) = R$$

Since $T_{\delta'}(i) = V_{\delta'}(i), \forall i$, the policy δ' is an optimal stationary policy.

Linear Programming

It can be shown that an optimal stationary policy for a maximization problem can be found by solving the following LP:

$$\min \sum_{j=1}^{N} V_j$$

$$V_i - \beta \sum_{j=1}^{N} p(j|i,d)V_j \geq r_{id}, \quad \forall i, \quad \forall d \in d(i)$$

For a minimization problem can be found by solving the following LP:

$$\max \sum_{j=1}^{N} V_j$$

$$V_i - \beta \sum_{j=1}^{N} p(j|i,d)V_j \leq r_{id}, \quad \forall i, \quad \forall d \in d(i)$$

Linear Programming

$$\min V_E + V_G + V_A + V_B$$

$$\begin{aligned}
&V_E \geq 100 + .9(.7V_E + .3V_G) && \text{(K in E)}\\
&V_G \geq 80 + .9(.7V_G + .3V_A) && \text{(K in G)}\\
&V_G \geq -100 + .9(.7V_E + .3V_G) && \text{(R in G)}\\
&V_A \geq 50 + .9(.6V_A + .4V_B) && \text{(K in A)}\\
&V_A \geq -100 + .9(.7V_E + .3V_G) && \text{(R in A)}\\
&V_B \geq 10 + .9V_B && \text{(K in B)}\\
&V_B \geq -100 + .9(.7V_E + .3V_G) && \text{(R in B)}
\end{aligned}$$

By solving the LP, we obtain $V_E = 690.23$, $V_G = 575.50$, $V_A = 492.35$ and $V_B = 490.23$.

Note that 1st, 2nd, 4th and 7th constraints are binding.

Value Iteration

$$V_t(i) = \max_{d \in D(i)} \left\{ r_{i,d} + \beta \sum_{j=1}^{N} p(j|i,d) V_{t-1}(j) \right\}, \quad t \geq 1$$

$$V_0(i) = 0$$

Let $d_t(i)$ be the decision that must be chosen during period 1 in state i to attain $V_t(i)$. For an MDP with finite state space and each $D(i)$ containing a finite number of elements, we have,

$$|V_t(i) - V(i)| \leq \frac{\beta^t}{1-\beta} \max_{i,d} |r_{id}|, \quad i = 1, \ldots, N$$

For an optimal stationary policy $\delta^*(i)$, we have

$$\lim_{t \to \infty} d_t(i) = \delta^*(i)$$

Value Iteration

Stage 1:

$$V_1(E) = 100 \quad (K)$$

$$V_1(G) = \max\begin{Bmatrix} 80 & (K) \\ -100 & (R) \end{Bmatrix} = 80$$

$$V_1(A) = \max\begin{Bmatrix} 50 & (K) \\ -100 & (R) \end{Bmatrix} = 50$$

$$V_1(B) = \max\begin{Bmatrix} 10 & (K) \\ -100 & (R) \end{Bmatrix} = 10$$

Value Iteration

Stage 2:

$$V_2(E) = 100 + .9[.7V_1(E) + .3V_1(G)] = 184.6 \quad (K)$$

$$V_2(G) = \max\begin{Bmatrix} 80 + .9[.7V_1(G) + .3V_1(A)] = 143.9 & (K) \\ -100 + .9[.7V_1(E) + .3V_1(G)] = -15.4 & (R) \end{Bmatrix} = 143.9$$

$$V_2(A) = \max\begin{Bmatrix} 50 + .9[.6V_1(A) + .4V_1(B)] = 80.6 & (K) \\ -100 + .9[.7V_1(E) + .3V_1(G)] = -15.4 & (R) \end{Bmatrix} = 80.6$$

$$V_2(B) = \max\begin{Bmatrix} 10 + .9V_1(B) = 19 & (K) \\ -100 + .9[.7V_1(E) + .3V_1(G)] = -15.4 & (R) \end{Bmatrix} = 19$$

Value Iteration

Observe that after two iterations of successive approximations, we have not yet come close to the actual values of $V(i)$ and have not found it optimal to replace even a bad machine. In general, if we want to ensure that all the $V_t(i)$ are within ϵ of the corresponding $V(i)$, we would perform t^* iterations of successive iterations where

$$\frac{\beta^{t^*}}{1-\beta}\max_{i,d}|r_{id}| < \epsilon$$

Maximizing Average Reward

$$\max \sum_{i=1}^{N} \sum_{d \in D(i)} \pi_{id} r_{id}$$

$$\sum_{i=1}^{N} \sum_{d \in D(i)} \pi_{id} = 1$$

$$\sum_{d \in D(j)} \pi_{jd} = \sum_{d \in D(i)} \sum_{i=1}^{N} \pi_{id} p(j|i,d), \quad \forall j$$

$$\pi_{id} \geq 0, \quad \forall i, d$$

Machine Replacement

$$\max 100\pi_{EK} + 80\pi_{GK} + 50\pi_{AK} + 10\pi_{BK} - 100(\pi_{GR} + \pi_{AR} + \pi_{BR})$$

$$\pi_{EK} + \pi_{GK} + \pi_{AK} + \pi_{BK} + \pi_{GR} + \pi_{AR} + \pi_{BR} = 1$$

$$\pi_{EK} = .7(\pi_{EK} + \pi_{GR} + \pi_{AR} + \pi_{BR})$$

$$\pi_{GK} + \pi_{GR} = .3(\pi_{GR} + \pi_{AR} + \pi_{BR} + \pi_{EK}) + .7\pi_{GK}$$

$$\pi_{AR} + \pi_{AK} = .3\pi_{GK} + .6\pi_{AK}$$

$$\pi_{BR} + \pi_{BK} = \pi_{BK} + .4\pi_{AK}$$

By solving, we obtain $z = 60$ and $\pi_{EK} = .35$, $\pi_{GK} = .50$, $\pi_{AR} = .15$.

The End