SEEM 3470: Dynamic Optimization and Applications **2013–14 Second Term**

Handout 8: Introduction to Stochastic Dynamic Programming

Instructor: Shiqian Ma March 10, 2014

Suggested Reading: Chapter 1 of Bertsekas, *Dynamic Programming and Optimal Control: Volume I (3rd Edition)*, Athena Scientific, 2005; Chapter 2 of Powell, *Approximate Dynamic Programming: Solving the Curse of Dimensionalty (2nd Edition)*, Wiley, 2010.

1 Introduction

So far we have focused on the formulation and algorithmic solution of deterministic dynamic programming problems. However, in many applications, there are random perturbations in the system, and the deterministic formulations may no longer be appropriate. In this handout, we will introduce some examples of stochastic dynamic programming problems and highlight their differences from the deterministic ones.

2 Examples of Stochastic Dynamic Programming Problems

2.1 Asset Pricing

Suppose that we hold an asset whose price fluctuates randomly. Typically, the price change between two successive periods is assumed to be independent of prior history. A question of fundamental interest is to determine the best time to sell the asset, and as a by–product, infer the value of the asset at the time of selling. To formulate this problem, let P_k be the price of the asset that is revealed in period k. Note that in period k', where $k' < k$, the value of P_k is a random variable. Now, in period k, after P_k is revealed, we have to make a decision x_k, for which there are only two choices:

$$x_k = \begin{cases} 1 & \text{sell the asset,} \\ 0 & \text{hold the asset.} \end{cases} \tag{1}$$

We also use S_k to indicate the state of our asset right after P_k is revealed but before we make the decision x_k, where

$$S_k = \begin{cases} 1 & \text{asset held,} \\ 0 & \text{asset sold.} \end{cases}$$

With this setup, our goal is to solve the following optimization problem:

$$\max_k \mathbb{E}\left[P_k\right]. \tag{2}$$

Let $\hat{K}$ be an optimal solution to the above problem. Then, by definition, $\hat{K}$ is the time at which the expected value of the asset, i.e., $\mathbb{E}\left[P_{\hat{K}}\right]$, is largest. Hence, we should sell the asset at time $\hat{K}$, which implies that $x_{\hat{K}} = 1$. We refer to $\hat{K}$ as the *optimal stopping time*.

Before we discuss how to find the optimal stopping time, it is instructive to understand what structures should it possess. Observe that it does not make sense for $\hat{K}$ to be a fixed number. Indeed, suppose for the sake of argument that $\hat{K}$ is a fixed number, say $\hat{K} = 3$. This means that

no matter what happens to the price of the asset in periods 1 to 3, you will sell it in period 3. Such a strategy is certainly counter–intuitive, because it totally ignores the price information revealed in periods 1 to 3. A more reasonable strategy is to let $\hat{K}$ *depend on the asset price and the state of the system.* As it turns out, this is one of the most important differences between deterministic and stochastic systems. In a deterministic system, the optimal controls in each period can be fixed at the beginning, i.e., before the system starts evolving. This is because the evolution of the system is deterministic and there is no new information as time progresses. However, in a stochastic system, there are random parameters whose values become known in each period. These new information should be taken into account when devising the optimal controls. Thus, the optimal control in each period should depend on the state and the realizations of the random parameters.

Returning to the asset pricing problem, in order to formalize the state and price dependence of the optimal stopping time, we let the control x_k in period k be given by $x_k = \mu_k(P_k, S_k)$, where $\mu_k(\cdot,\cdot)$ is the policy in period k, P_k is the price of the asset in period k, and S_k is the state in period k. By definition, if $S_k = 0$, then we no longer hold the asset, and we have $x_k = \mu_k(P_k, 0) = 0$. If $S_k = 1$, then $x_k = \mu_k(P_k, 1)$ can be either 0 or 1, where $x_k = \mu_k(P_k, 1) = 1$ means we sell the asset in period k, and $x_k = \mu_k(P_k, 1) = 0$ means we hold the asset in period k (see (1)). Now, observe that only one of the controls $x_0, x_1, \ldots$ can equal to 1 (the asset can only be sold once). Hence, we can reformulate the problem of finding the optimal stopping time, i.e., problem (2), as follows:

$$\max_{\mu_0,\mu_1,\ldots} \mathbb{E}\left[\sum_{k=0}^{\infty} \mu_k(P_k, S_k) \cdot P_k\right]. \tag{3}$$

In other words, we are looking for the set of policies $\{\mu_0, \mu_1, \ldots\}$ that can maximize the expected price of the asset.

In general, problem (3) is difficult to handle, since $\mu_0, \mu_1, \ldots$ are functions. To simplify the problem, we may consider restricting our attention to functions of a certain type. For instance, we may require $\mu_0, \mu_1, \ldots$ to take the form

$$\mu_k^P(P_k, S_k) = \begin{cases} 1 & \text{if } P_k \geq P \text{ and } S_k = 1, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where $P > 0$ is a fixed number. In words, the policy in (4) says that we will sell the asset in period k if we still hold it in period k and the price P_k exceeds the threshold P. The upshot of using policies of the form in (4) is that they are parametrized by a single number P, and the optimization problem

$$\max_P \mathbb{E}\left[\sum_{k=0}^{\infty} \mu_k^P(P_k, S_k) \cdot P_k\right] \tag{5}$$

should be simpler than problem (3) because it involves a single decision variable P rather than general functions $\mu_0, \mu_1, \ldots$. However, the optimal value of problem (5) will generally be lower than that of problem (3) (i.e., the maximum expected selling price given in (5) will be lower than that given in (3)), because we only consider a special class of policies in (5). Thus, an important problem is to determine when would the optimal policies for (3) take the form (4). We shall return to this question later in the course.

2.2 Batch Replenishment

Consider a single type of resource that is being stored, say, in a warehouse and consumed over time. As the resource level runs low, we need to replenish the warehouse. However, there is economy of scale when doing the replenishment. Specifically, it is cheaper on average to increase the resource level in batches. To model this situation, let

- S_k be the resource level at the beginning of period k,
- x_k be the resource acquired at the beginning of period k to be used between periods k and $k+1$,
- W_k be the (random) demand between periods k and $k+1$, and
- N be the length of the planning horizon.

The transition function is given by

$$S_{k+1} = \max\{0, S_k + x_k - W_k\}.$$

In words, the total resource available at the beginning of period k, namely, $S_k + x_k$, is used to satisfy the random demand W_k, and we assume that the unsatisfied demand is lost.

Now, the cost incurred in period k is given by

$$\Lambda(S_k, x_k, W_k) = f \cdot \mathbb{I}(x_k > 0) + p \cdot x_k + h \cdot \max\{0, S_k + x_k - W_k\} + u \cdot \max\{0, W_k - S_k - x_k\},$$

where

$$\mathbb{I}(x_k > 0) = \begin{cases} 1 & \text{if } x_k > 0, \\ 0 & \text{if } x_k = 0 \end{cases}$$

is the indicator of the event $x_k > 0$, f is the fixed ordering cost, p is the unit ordering cost, h is the unit holding cost, and u is the penalty for each unit of unsatisfied demand.

In general, the optimal control in each period will depend on the state in that period. Hence, we are interested in finding a set of policies $\{\mu_0, \mu_1, \ldots, \mu_N\}$ to minimize the total cost, i.e.,

$$\min_{\mu_0,\ldots,\mu_N} \mathbb{E}\left[\sum_{k=0}^{N-1} \Lambda(S_k, \mu_k(S_k), W_k)\right]. \tag{6}$$

Note that problem (6) essentially asks for two decisions, namely, when to replenish and how much to replenish. Again, it may be difficult to deal with arbitrary policies. To simplify the problem, we may consider, for instance, the following class of policies:

$$\mu_k^{Q,q}(S_k) = \begin{cases} 0 & \text{if } S_k \geq q, \\ Q - S_k & \text{if } S_k < q. \end{cases} \tag{7}$$

The policies in (7) are parametrized by a pair of numbers (Q, q). In words, it says that if the resource level is larger than q, then we do not replenish. Otherwise, we replenish up to the level Q. Then, we may consider the following optimization problem:

$$\min_{Q,q} \mathbb{E}\left[\sum_{k=0}^{N-1} \Lambda(S_k, \mu_k^{Q,q}(S_k), W_k)\right]. \tag{8}$$

Problem (8) is simpler than problem (6) in the sense that it only involves the two decision variables Q, q. However, it is important to determine whether the optimal policies for problem (6) have the same structure as those given in (7).

3 The Dynamic Programming (DP) Algorithm Revisited

After seeing some examples of stochastic dynamic programming problems, the next question we would like to tackle is how to solve them. Towards that end, it is helpful to recall the derivation of the DP algorithm for *deterministic* problems. Suppose that we have an N–stage deterministic DP problem, and suppose that at the beginning of period k (where $0 \leq k \leq N-1$), we are in state S_k. Now, note that the next state S_{k+1} is uniquely determined by the state S_k, the control x_k, and the parameter w_k in period k, i.e., $S_{k+1} = \Gamma_k(S_k, x_k, w_k)$, because w_k is deterministic. Thus, if we fix the control x_k, then we have

$$\begin{aligned} & \text{optimal cost to go from state } S_k \text{ to the terminal state } t \text{ by using control } x_k \qquad (9) \\ = \;& \text{optimal cost to go from state } S_k \text{ to the terminal state } t \text{ through state } S_{k+1} = \Gamma_k(S_k, x_k, w_k) \\ = \;& \Lambda_k(S_k, x_k, w_k) + \text{optimal cost to go from state } S_{k+1} = \Gamma_k(S_k, x_k, w_k) \text{ to the terminal state } t, \end{aligned}$$

where $\Lambda_k(S_k, x_k, w_k)$ is the cost to go from S_k to $S_{k+1} = \Gamma_k(S_k, x_k, w_k)$; see Figure 1.

Figure 1: Illustration of the deterministic DP Algorithm. Given the current state $S_k = i$ and control $x_k = x$, the next state $S_{k+1} = j$ is uniquely determined by the transition function $S_{k+1} = \Gamma_k(S_k, x_k, w_k)$, and the cost incurred is $\Lambda_k(S_k, x_k, w_k)$.

In particular, if we let

$$\begin{aligned} J_k(S_k) &= \text{optimal cost to go from state } S_k \text{ to the terminal state } t \\ &= \min_{x_k} \big\{ \text{optimal cost to go from state } S_k \text{ to the terminal state } t \text{ by using control } x_k \big\}, \end{aligned}$$

then we see from (9) that

$$J_k(S_k) = \min_{x_k} \big\{ \Lambda_k(S_k, x_k, w_k) + J_{k+1}(\Gamma_k(S_k, x_k, w_k)) \big\} \quad \text{for } k = 0, 1, \ldots, N-1, \qquad (10)$$

with the boundary condition given by

$$J_N(S_N) = \Lambda_N(S_N). \qquad (11)$$

The reader should now recognize that (10) and (11) are precisely the recursion equations in the DP algorithm.

As it turns out, the derivation of the DP algorithm for *stochastic* problems is largely similar. The only difference is that the next state S_{k+1} is no longer uniquely given by the state S_k, the control x_k and the (random) parameter W_k in period k. (Here, we capitalize W in W_k to indicate

the fact that W_k is now a random variable.) Instead, we assume that the next state S_{k+1} is specified by a probability distribution:

$$p_{ij}(x) = \Pr(S_{k+1} = j \,|\, S_k = i, x_k = x). \tag{12}$$

One way to understand (12) is to observe that it specifies the transition probabilities of a Markov chain for each fixed control $x_k = x$. Thus, we can use the theory of Markov chains to study this type of stochastic DP.

Now, the analog of (9) in the context of stochastic DP becomes

$$\begin{aligned}
&\mathbb{E}\left[\text{optimal cost to go from } S_k = i \text{ to } t \text{ by using } x_k = x\right] \\
= &\sum_{p=1}^{l} \mathbb{E}\left[\text{optimal cost to go from } S_k = i \text{ to } t \text{ through } S_{k+1} = j_p\right] \times \Pr(S_{k+1} = j_p \,|\, S_k = i, x_k = x) \\
= &\sum_{p=1}^{l} p_{i,j_p}(x) \cdot \mathbb{E}\left[\Lambda_k(i, x, W_k) + \text{optimal cost to go from } S_{k+1} = j_p \text{ to } t\right] \\
= &\sum_{p=1}^{l} p_{i,j_p}(x) \cdot \left\{ \mathbb{E}\left[\Lambda_k(i, x, W_k)\right] + \mathbb{E}\left[\text{optimal cost to go from } S_{k+1} = j_p \text{ to } t\right] \right\}, \\
= &\mathbb{E}\left[\Lambda_k(i, x, W_k)\right] + \sum_{p=1}^{l} p_{i,j_p}(x) \cdot \mathbb{E}\left[\text{optimal cost to go from } S_{k+1} = j_p \text{ to } t\right], \qquad (13)
\end{aligned}$$

where we assume that

$$\sum_{p=1}^{l} p_{i,j_p}(x) = 1,$$

i.e., if the control in period k is $x_k = x$, then $S_{k+1} \in \{j_1, \ldots, j_l\}$; see Figure 2.

Hence, if we let

$$J_k(S_k) = \mathbb{E}\left[\text{optimal cost to go from } S_k \text{ to } t\right],$$

then we deduce from (13) that

$$J_k(S_k) = \min_{x_k} \left\{ \mathbb{E}\left[\Lambda(S_k, x_k, W_k)\right] + \sum_{p=1}^{l} p_{S_k,j_p}(x_k) \cdot J_{k+1}(j_p) \right\}, \tag{14}$$

with the boundary condition given by

$$J_N(S_N) = \Lambda_N(S_N). \tag{15}$$

In particular, the stochastic DP algorithm is given by (14) and (15).

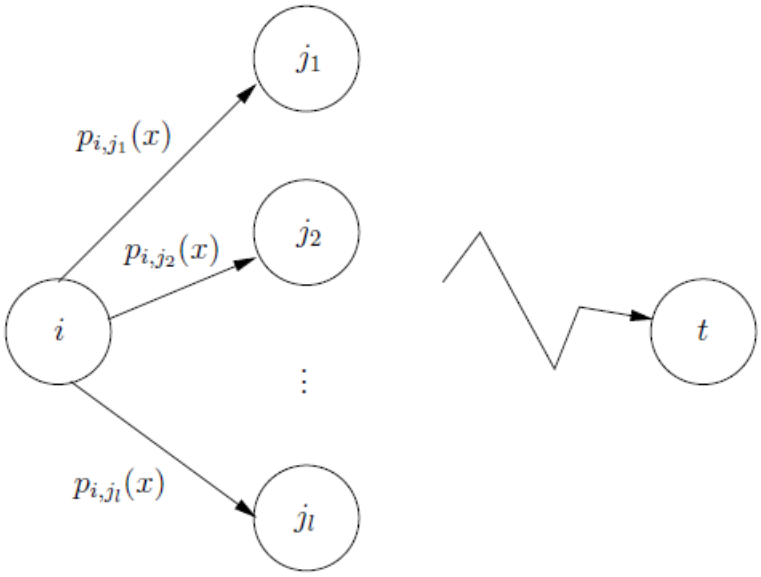


Figure 2: Illustration of the stochastic DP Algorithm. Given the current state $S_k = i$ and control $x_k = x$, the next state S_{k+1} is random and is determined by the transition probabilities $p_{ij}(x) = \Pr(S_{k+1} = j \mid S_k = i, x_k = x)$.

3.1 Example: Stochastic Inventory Problem

Consider an inventory system, where at the beginning of period k, the inventory level is S_k, and we can order x_k units of goods. The available units of goods are then used to serve a random demand W_k, and the amount of inventory carried over to the next period is $S_{k+1} = \max\{0, S_k + x_k - W_k\}$. We assume that S_k, x_k, W_k are non–negative integers, and that the random demand W_k follows the probability distribution

$$\Pr(W_k = 0) = 0.1, \quad \Pr(W_k = 1) = 0.7, \quad \Pr(W_k = 2) = 0.2 \quad \text{for all } k = 0, 1, \ldots, N-1.$$

The cost incurred in period k is

$$\Lambda_k(S_k, x_k, W_k) = (S_k + x_k - W_k)^2 + x_k.$$

Furthermore, there is a storage constraint in each period k, which is given by $S_k + x_k \leq 2$. The terminal cost is given by $\Lambda_N(S_N) = 0$.

Now, consider a 2–period problem, i.e., $N = 2$, where we assume that $S_0 = 0$, and our goal is to find the optimal ordering quantities x_0 and x_1. This can be done by applying the stochastic DP algorithm (14)–(15). First, observe that because of the storage constraint, we have $S_k \in \{0, 1, 2\}$ for all k. Moreover, by the given terminal condition, we have

$$J_2(0) = J_2(1) = J_2(2) = 0.$$

Next, using (14), we consider

$$\begin{aligned} J_1(S_1) &= \min_{\substack{0\le x_1\le 2-S_1 \\ x_1 \text{ integer}}} \left\{ \mathbb{E}\left[\Lambda_1(S_1, x_1, W_1)\right] + \sum_{p=0}^{2} p_{S_1,p}(x_1)\cdot J_2(p) \right\} \\ &= \min_{\substack{0\le x_1\le 2-S_1 \\ x_1 \text{ integer}}} \mathbb{E}\left[(S_1 + x_1 - W_1)^2 + x_1\right] \\ &= \min_{\substack{0\le x_1\le 2-S_1 \\ x_1 \text{ integer}}} \left[x_1 + (0.1)\times(S_1+x_1)^2 + (0.7)\times(S_1+x_1-1)^2 + (0.2)\times(S_1+x_1-2)^2\right]. \end{aligned}$$

To find $J_1(S_1)$, we can simply do an exhaustive search, since S_1 can only equal to $0, 1$ or 2. Now, we compute

$$\begin{aligned} J_1(0) &= \min_{\substack{0\le x_1\le 2 \\ x_1 \text{ integer}}} \left[x_1 + (0.1)\times x_1^2 + (0.7)\times(x_1-1)^2 + (0.2)\times(x_1-2)^2\right] \\ &= \min_{\substack{0\le x_1\le 2 \\ x_1 \text{ integer}}} \left[x_1^2 - (1.2)\times x_1 + 1.5\right] \\ &= 1.3, \end{aligned}$$

and the optimal control x_1^* when $S_1 = 0$ is given by $x_1^* = \mu_1(0) = 1$. Similarly, we have

$$J_1(1) = \min_{\substack{0\le x_1\le 1 \\ x_1 \text{ integer}}} \left[x_1^2 - (0.2)\times x_1 + 0.3\right] = 0.3 \quad \text{with } x_1^* = \mu_1(1) = 0,$$

$$J_1(2) = (0.1)\times 4 + (0.7)\times 1 + (0.2)\times 0 = 1.1 \quad \text{with } x_1^* = \mu_1(2) = 0.$$

Now, using (14) again, we have

$$J_0(S_0) = \min_{\substack{0\le x_0\le 2-S_0 \\ x_0 \text{ integer}}} \left\{ \mathbb{E}\left[\Lambda_0(S_0, x_0, W_0)\right] + \sum_{p=0}^{2} p_{S_0,p}(x_0)\cdot J_1(p) \right\}.$$

By assumption, $S_0 = 0$. Thus, the above equation simplifies to

$$\begin{aligned} J_0(0) &= \min_{\substack{0\le x_0\le 2 \\ x_0 \text{ integer}}} \left\{ x_0 + (0.1)\times x_0^2 + (0.7)\times(x_0-1)^2 + (0.2)\times(x_0-2)^2 + \sum_{p=0}^{2} p_{0,p}(x_0)\cdot J_1(p) \right\} \\ &= \min_{\substack{0\le x_0\le 2 \\ x_0 \text{ integer}}} \underbrace{\left\{ x_0^2 - (1.2)\times x_0 + 1.5 + \sum_{p=0}^{2} p_{0,p}(x_0)\cdot J_1(p) \right\}}_{f(x_0)}. \end{aligned}$$

Now, observe that

$$p_{0,0}(0) = \Pr(S_1 = \max\{0, 0 - W_0\} = 0 \,|\, S_0 = 0,\, x_0 = 0) = 1, \quad p_{0,1}(0) = p_{0,2}(0) = 0,$$

$$p_{0,0}(1) = \Pr(S_1 = \max\{0, 1 - W_0\} = 0 \,|\, S_0 = 0, x_0 = 1) = 0.9, \quad p_{0,1}(1) = 0.1, \quad p_{0,2}(1) = 0,$$

$$p_{0,0}(2) = \Pr(S_1 = \max\{0, 2 - W_0\} = 0 \,|\, S_0 = 0, x_0 = 2) = 0.2, \quad p_{0,1}(2) = 0.7, \quad p_{0,2}(2) = 0.1.$$

Hence, we have

$$f(0) = 0 - (1.2) \times 0 + 1.5 + \sum_{p=0}^{2} p_{0,p}(0) \cdot \mathbb{E}\left[J_1(p)\right] = 1.5 + J_1(0) = 2.8,$$

$$f(1) = 1 - (1.2) \times 1 + 1.5 + \sum_{p=0}^{2} p_{0,p}(1) \cdot \mathbb{E}\left[J_1(p)\right] = 1.3 + (0.9) \times J_1(0) + (0.1) \times J_1(1) = 2.5,$$

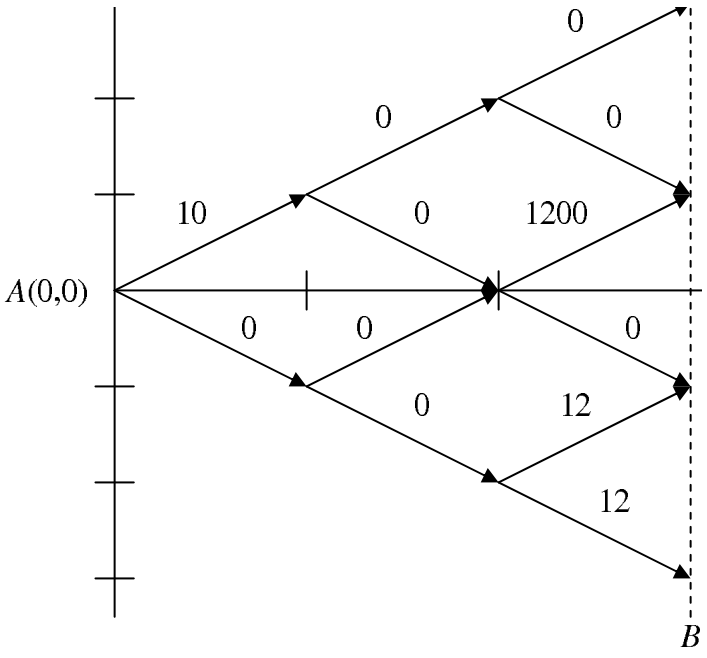
$$f(2) = 4 - (1.2) \times 2 + 1.5 + \sum_{p=0}^{2} p_{0,p}(2) \cdot \mathbb{E}\left[J_1(p)\right] = 3.68.$$

In particular, we conclude that

$$J_0(0) = 2.5 \quad \text{with } x_0^* = \mu_0(0) = 1.$$

3.2 Example: Stochastic Shortest Path

Example: Find an optimal policy to go from $A(0,0)$ to the line B with minimum expected cost where the probability of succeeding at each vertex is $p = 0.75$. ⊠



Let $L((x_1, y_1), (x_2, y_2))$ = the cost incurred when traveling from (x_1, y_1) to (x_2, y_2), where $x_2 = x_1 + 1$.

Stage: x-coordinate, i.e., $x = 0, 1, 2, 3$.

State: $y_x = y$-coordinate at stage x:

$$y_0 = 0, \quad y_1 = 1, -1, \quad y_2 = 2, 0, -2, \quad y_3 = 3, 1, -1, -3.$$

Decision: $d_x(y_x)$ = move direction at state y_x of stage x.
$d_x(y_x) = U$, D, for all y_x, x.

Transition Equation: $y_{x+1} = \begin{cases} y_x + 1 \text{ with probability } p, & \text{if } d_x(y_x) = U \\ y_x - 1 \text{ with probability } 1-p, & \text{if } d_x(y_x) = U \\ y_x + 1 \text{ with probability } 1-p, & \text{if } d_x(y_x) = D \\ y_x - 1 \text{ with probability } p, & \text{if } d_x(y_x) = D \end{cases}$

Recursive Relation and Boundary Conditions:

$f_x(y_x, d_x(y_x))$
= minimum expected cost from state y_x of stage x to the line B, given that $d_x(y_x)$ is the decision at state y_x of stage x

$$= \begin{cases} p[L((x,y_x),(x+1,y_x+1)) + f^*_{x+1}(y_x+1)] \\ \quad + (1-p)[L((x,y_x),(x+1,y_x-1)) + f^*_{x+1}(y_x-1)], \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } d_x(y_x) = U, \\ (1-p)[L((x,y_x),(x+1,y_x+1)) + f^*_{x+1}(y_x+1)] \\ \quad + p[L((x,y_x),(x+1,y_x-1)) + f^*_{x+1}(y_x-1)], \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } d_x(y_x) = D. \end{cases}$$

$f^*_x(y_x)$
= minimum expected cost from state y_x of stage x to the line B
$= \min\limits_{d_x(y_x)} f_x(y_x, d_x(y_x)), \quad$ for $x = 0, 1, 2$

$f^*_3(3) = 0, \quad f^*_3(1) = 0, \quad f^*_3(-1) = 0, \quad f^*_3(-3) = 0.$

Goal: $f^*_0(0)$.

Stage 3:

y_3	$f^*_3(y_3)$
3	0
1	0
-1	0
-3	0

Stage 2:

	$d_2(y_2) = U$	$d_2(y_2) = D$		
y_2	$f_2(y_2, U)$	$f_2(y_2, D)$	$f^*_2(y_2)$	$d^*_2(y_2)$
2	0	0	0	U or D
0	900	300	300	D
-2	12	12	12	U or D

Stage 1:

	$d_1(y_1) = U$	$d_1(y_1) = D$		
y_1	$f_1(y_1, U)$	$f_1(y_1, D)$	$f^*_1(y_1)$	$d^*_1(y_1)$
1	75	225	75	U
-1	228	84	84	D

Stage 0:

y_0	$d_0(y_0) = U$ $f_0(y_0, U)$	$d_0(y_0) = D$ $f_0(y_0, D)$	$f_0^*(y_0)$	$d_0^*(y_0)$
0	84.75	84.25	84.25	D

Answers: The minimum expected cost = 84.25.