Exercise 2.7

a) The proof of termination for this algorithm follows exactly that for the original algorithm. Each time a node j enters the OPEN list, its label is decreased and becomes equal to the length of some path from s to j. Although arc lengths are no longer necessarily nonnegative, cycles lengths are. Therefore, since each path can be decomposed into a path with no repeated nodes (there is a finite number of distinct such paths) plus a (possibly empty) set of cycles (which have a nonnegative length), the number of distinct lengths of paths from s to j that are smaller than any given number is finite. Therefore, there can be only a finite number of label reductions and the algorithm must terminate.

Let $(s, j_1, j_2, \ldots, j_k, t)$ be a shortest path from s to t and let d^* be the corresponding shortest distance. We will show that the value of UPPER upon termination must be equal to d^*. Indeed, each subpath $(s, j_1, \ldots, j_m)$, $m = 1, \ldots, k$, of the shortest path must be a shortest path from s to j_m. Let $d^*_{j_m}$ be the corresponding shortest distance from s to j_m, $m = 1, \ldots, k$. If the value of UPPER is larger than d^* at termination, the same must be true throughout the algorithm, and therefore UPPER will also be larger than the length of all the paths $(s, j_1, \ldots, j_m)$ plus the underestimate u_{j_m}, $m = 1, \ldots, k$, throughout the algorithm, i.e.,

$$d^*_{j_m} < \text{UPPER} - u_{j_m}, \qquad m = 1, \ldots, k. \tag{$*$}$$

Using this relation for $m = k$, it follows that j_k, the last node prior to t on the shortest path, will never enter the OPEN list with d_{j_k} equal to the shortest distance $d^*_{j_k}$, since when this happens UPPER would be set to $d^* = d^*_{j_k} + a_{j_k t}$ in step 2 immediately following the next time node j_k is examined by the algorithm in step 2. Going backwards one step and using Eq. (*), node j_{k-1} will never enter the OPEN list with $d_{j_{k-1}}$ equal to the shortest distance $d^*_{j_{k-1}}$. Proceeding backward similarly, we conclude that j_1 never enters the OPEN list with d_{j_1} equal to the shortest distance $d^*_{j_1}$ [which is equal to the length of the arc (s, j_1)]. This happens, however, at the first iteration of the algorithm, obtaining a contradiction. It follows that at termination, UPPER will be equal to the shortest distance from s to t. **Q.E.D.**

b) In the case where all arcs have nonnegative lengths, an underestimate of the shortest distance from any node to the destination node is clearly 0. Letting $u_j = 0$ from all nodes j, we see that the algorithm described reduces to the algorithm of Section 2.3.1.

Exercise 1.2

The algorithm takes the form:

$$J_k(x_k) = \min_{\substack{0\le u_k\le 2-x_k\\ u_k=0,1,2}} \underset{w_k}{E}\{u_k + (x_k + u_k - w_k)^2 + J_{k+1}(\max(0, x_k + u_k - w_k))\}.$$

For the initial state $x_0 = 1$, we have

$$J_0(1) = \min_{u_0=0,1} \underset{w_0}{E}\{u_0 + (1 + u_0 - w_0)^2 + J_1(\max(0, 1 + u_0 - w_0))\},$$

$$u_0 = 0 : E\{\cdot\} = 0.1(1 + J_1(1)) + 0.7 \cdot J_1(0) + 0.2(1 + J_1(0)) = 2.7$$
$$u_0 = 1 : E\{\cdot\} = 1 + 0.1(4 + J_1(2)) + 0.7(1 + J_1(1)) + 0.2 \cdot J_1(0) = 3.818$$

$$J_0(1) = 2.7, \quad \mu_0^*(1) = 0$$

For the initial state $x_0 = 2$, the only admissible control is $u_0 = 0$, and we have

$$\begin{aligned} J_0(2) &= \underset{w_0}{E}\{(2 - w_0)^2 + J_1(\max(0, 2 - w_0))\} \\ &= 0.1(4 + J_1(2)) + 0.7(1 + J_1(1)) + 0.2 \cdot J_1(0) \\ &= 2.818 \end{aligned}$$

$$J_0(2) = 2.818, \quad \mu_0^*(2) = 0.$$

Exercise 1.3

Except for the first week, there are only two permissible states R (= running) and B (= broken down). For $x_k = R$, there are only two permissible control values, namely m (= perform maintenance) and n (= no maintenance). For $x_k = B$, there are two different permissible control values, namely r (= repair), and ℓ (= replace). For the first week, we're in state N (new machine, and the machine is guaranteed to run through its first week of operation. Number the weeks from 0 to 3. Using the given probabilities we have:

Week 3: For

$$x_3 = R \qquad \begin{cases} \text{for } u_3 = m, & \text{expected profit} = 0.6 \cdot 100 - 20 = 40 \\ \text{for } u_3 = n, & \text{expected profit} = 0.3 \cdot 100 = 30 \end{cases}$$

Thus, $J_3(R) = \max\{40, 30\} = 40$ and $\mu_3^*(R) = m$. For

$$x_3 = B \qquad \begin{cases} \text{for } u_3 = r, & \text{expected profit} = 0.6 \cdot 100 - 40 = 20 \\ \text{for } u_3 = \ell, & \text{expected profit} = 100 - 150 = -50 \end{cases}$$

Thus, $J_3(B) = \max\{20, -50\} = 20$ and $\mu_3^*(B) = r$.

Week 2: For

$$x_2 = R \qquad \begin{cases} \text{for } u_2 = m, & \text{expected profit} = 0.6[100 + J_3(R)] + 0.4 \cdot J_3(B) - 20 = 72 \\ \text{for } u_2 = n, & \text{expected profit} = 0.3[100 + J_3(R)] + 0.7 \cdot J_3(B) = 56 \end{cases}$$

Thus, $J_2(R) = \max\{72, 56\} = 72$ and $\mu_2^*(R) = m$. For

$$x_2 = B \qquad \begin{cases} \text{for } u_2 = r, & \text{expected profit} = 0.6[100 + J_3(R)] + 0.4 J_3(B) - 40 = 52 \\ \text{for } u_2 = \ell, & \text{expected profit} = 100 + J_3(R) - 150 = -10 \end{cases}$$

Thus, $J_2(B) = \max\{52, -10\} = 52$ and $\mu_2^*(B) = r$.

Week 1: For

$$x_2 = R \qquad \begin{cases} \text{for } u_1 = m, & \text{expected profit} = 0.6[100 + J_2(R)] + 0.4 \cdot J_2(B) - 20 = 104 \\ \text{for } u_1 = n, & \text{expected profit} = 0.3[100 + J_2(R)] + 0.7 J_2(B) = 88 \end{cases}$$

Thus, $J_1(R) = \max\{104, 88\} = 104$ and $\mu_1^*(R) = m$. For

$$x_1 = B \qquad \begin{cases} \text{for } u_2 = r, & \text{expected profit} = 0.6[100 + J_2(R)] + 0.4 \cdot J_2(B) - 40 = 84 \\ \text{for } u_1 = \ell, & \text{expected profit} = 100 + J_2(R) - 150 = 22 \end{cases}$$

Thus, $J_1(B) = \max\{84, 22\} = 84$ and $\mu_1^*(B) = r$.

Week 0: We start with a new machine, which is guaranteed to run through its first week of operation. Thus $J_0(N) = 100 + J_1(R) = 204$.

Conclusion: The profit-maximizing policy is to always maintain a running machine and always repair a broken one. The corresponding expected profit is \$204.

Exercise 1.22

We consider the problem of placing N points on a circle, so that the resulting polygon has maximal perimeter. As the circle is traversed in the clockwise direction, we number sequentially the encountered points as $x_1, x_2, \ldots, x_N$. Without loss of generality, we assume that x_1 is at 12 o' clock on the circle. For any point x on the circle, we denote by $\phi \in [0, 2\pi]$ the angle between x_1 and x (measured clockwise), and we let $x = x(\phi)$ (in other words, we parameterize every point on the circle with the angle ϕ). We denote by $J_k(\phi)$ the maximal sum of chords with first vertex $x(\phi)$, last vertex x_1 and $N - k$ additional points on the subarc that lies (clockwise) between x and x_1 (thus, we have $N - k + 1$ chords). Finally, without loss of generality, we assume that the radius of the circle is 1, so that the length of a chord that has as vertices two points on the circle is $2\sin(u/2)$, where u is the angle with respect to the center.

By viewing as state the angle ϕ_k between x_1 and x_k (notice that, by definition, $0 \leq \phi_k \leq 2\pi$), and as control the angle u_k between x_k and x_{k+1}, we obtain the following DP algorithm

$$J_k(\phi_k) = \max_{0 \leq u_k \leq 2\pi - \phi_k} \left[2\sin(u_k/2) + J_{k+1}(\phi_k + u_k) \right], \qquad k = 1, \ldots, N-1. \tag{1}$$

$$J_k(\phi_k) = \max_{0 \leq u_k \leq 2\pi - \phi_k} H_k(u_k, \phi_k), \tag{4}$$

where

$$H_k(u_k, \phi_k) = 2\sin\frac{u_k}{2} + 2(N-k)\,\sin\left(\frac{2\pi - \phi_k - u_k}{2(N-k)}\right). \tag{5}$$

It can be verified that for a fixed ϕ_k and in the range $0 \leq u_k \leq 2\pi - \phi_k$, the function $H_k(\cdot, \phi_k)$ is concave (its second derivative is negative) and its derivative is 0 at the point $u_k^* = (2\pi - \phi_k)/(N - k + 1)$ which must therefore be its unique maximum. Substituting this value of u_k^* in Eqs. (4) and (5), we obtain

$$J_k(\phi_k) = 2\sin\left(\frac{2\pi - \phi_k}{2(N-k+1)}\right) + 2(N-k)\,\sin\left(\frac{2\pi - \phi_k - \frac{2\pi - \phi_k}{N-k+1}}{2(N-k)}\right) = 2(N-k+1)\,\sin\left(\frac{2\pi - \phi_k}{2(N-k+1)}\right),$$

and the induction is complete.

Thus, given an optimally placed point x_k with corresponding angle ϕ_k, the next point x_{k+1} is obtained by advancing clockwise by $(2\pi - \phi_k)/(N - k + 1)$. This process, when started at x_1, yields as the optimal solution an equally spaced placement of the points on the circle.

Exercise 2.1

The DP algorithm is:

$$J_{N-1}(i) = a_{it}$$

$$J_k(i) = \min_{j=1,\dots,N}\{a_{ij} + J_{k+1}(j)\} \quad k = 0, 1, \dots, N-2$$

where for this problem $t = 6$ and $N = 5$.

4th Stage:

$$\begin{aligned}
J_4(1) &= a_{16} = 8 && \text{path} = \{1 \to 6\}\\
J_4(2) &= a_{26} = \infty && \text{path} = \{2 \to 6\}\\
J_4(3) &= a_{36} = 9 && \text{path} = \{3 \to 6\}\\
J_4(4) &= a_{46} = 2 && \text{path} = \{4 \to 6\}\\
J_4(5) &= a_{56} = 5 && \text{path} = \{5 \to 6\}
\end{aligned}$$

3th Stage:

$$\begin{aligned}
J_3(1) &= \min\{a_{11} + J_4(1), a_{12} + J_4(2), a_{13} + J_4(3), a_{14} + J_4(4), a_{15} + J_4(5)\}\\
&= \min\{0 + 8, \infty, 1 + 9, 5 + 2, \infty\} = 7 && \text{path} = \{1 \to 4 \to 6\}\\
J_3(2) &= \min\{a_{21} + J_4(1), a_{22} + J_4(2), a_{23} + J_4(3), a_{24} + J_4(4), a_{25} + J_4(5)\}\\
&= \min\{\infty, \infty, \infty, 1 + 2, \infty\} = 3 && \text{path} = \{2 \to 4 \to 6\}\\
J_3(3) &= \min\{a_{31} + J_4(1), a_{32} + J_4(2), a_{33} + J_4(3), a_{34} + J_4(4), a_{35} + J_4(5)\}\\
&= \min\{\infty, \infty, 0 + 9, \infty, 5 + 5\} = 9 && \text{path} = \{3 \to 6\}\\
J_3(4) &= \min\{a_{41} + J_4(1), a_{42} + J_4(2), a_{43} + J_4(3), a_{44} + J_4(4), a_{45} + J_4(5)\}\\
&= \min\{\infty, \infty, \infty, 0 + 2, \infty\} = 2 && \text{path} = \{4 \to 6\}\\
J_3(5) &= \min\{a_{51} + J_4(1), a_{52} + J_4(2), a_{53} + J_4(3), a_{54} + J_4(4), a_{55} + J_4(5)\}\\
&= \min\{\infty, \infty, 0 + 9, \infty, 0 + 5\} = 5 && \text{path} = \{5 \to 6\}
\end{aligned}$$

2nd Stage:

$$\begin{aligned}
J_2(1) &= \min\{a_{11} + J_3(1), a_{12} + J_3(2), a_{13} + J_3(3), a_{14} + J_4(4), a_{15} + J_3(5)\}\\
&= \min\{0 + 7, 4 + 3, 1 + 9, 5 + 2, \infty\} = 7 && \text{path} = \{1 \to 4 \to 6 \text{ or } 1 \to 2 \to 4 \to 6\}\\
J_2(2) &= \min\{a_{21} + J_3(1), a_{22} + J_3(2), a_{23} + J_3(3), a_{24} + J_3(4), a_{25} + J_3(5)\}\\
&= \min\{\infty, 0 + 3, \infty, 1 + 2, \infty\} = 3 && \text{path} = \{2 \to 4 \to 6\}\\
J_2(3) &= \min\{a_{31} + J_3(1), a_{32} + J_3(2), a_{33} + J_3(3), a_{34} + J_3(4), a_{35} + J_3(5)\}\\
&= \min\{\infty, \infty, 0 + 9, \infty, 5 + 5\} = 9 && \text{path} = \{3 \to 6\}\\
J_2(4) &= \min\{a_{41} + J_3(1), a_{42} + J_3(2), a_{43} + J_3(3), a_{44} + J_3(4), a_{45} + J_3(5)\}\\
&= \min\{\infty, \infty, \infty, 0 + 2, \infty\} = 2 && \text{path} = \{4 \to 6\}\\
J_2(5) &= \min\{a_{51} + J_3(1), a_{52} + J_3(2), a_{53} + J_3(3), a_{54} + J_3(4), a_{55} + J_3(5)\}\\
&= \min\{\infty, \infty, 0 + 9, \infty, 0 + 5\} = 5 && \text{path} = \{5 \to 6\}
\end{aligned}$$

Since the shortest distances of paths with 2 arcs are the same as those of paths with 3 arcs, additional arcs will not create shorter paths. The shortest paths are therefore those found in the second stage.

Exercise 2.7

a) The proof of termination for this algorithm follows exactly that for the original algorithm. Each time a node j enters the OPEN list, its label is decreased and becomes equal to the length of some path from s to j. Although arc lengths are no longer necessarily nonnegative, cycles lengths are. Therefore, since each path can be decomposed into a path with no repeated nodes (there is a finite number of distinct such paths) plus a (possibly empty) set of cycles (which have a nonnegative length), the number of distinct lengths of paths from s to j that are smaller than any given number is finite. Therefore, there can be only a finite number of label reductions and the algorithm must terminate.

b) In the case where all arcs have nonnegative lengths, an underestimate of the shortest distance from any node to the destination node is clearly 0. Letting $u_j = 0$ from all nodes j, we see that the algorithm described reduces to the algorithm of Section 2.3.1.

Exercise 2.9

We will transform the problem to a (standard) shortest path problem in an expanded graph that is constructed from the given graph. Let I be the set of nodes of the original graph. The expanded graph has nodes $(i,0),(i,1),\ldots,(i,N)$, where i ranges over the node set I of the original graph. The meaning of being in node (i,m), $m=1,\ldots,N$, is that we are at node i and have already successively visited the sets $T_1,\ldots,T_m$, but not the sets $T_{m+1},\ldots,T_N$. The meaning of being in node $(i,0)$ is that we are at node i and have not yet visited any node in the set T_1.

The arcs of the expanded graph are constructed as follows: For each arc (i,j) of the original graph, with length a_{ij}, introduce for $m=0,\ldots,N-1$, in the expanded graph an arc of length a_{ij} that goes from (i,m) to (j,m) if $j\notin T_{m+1}$, and goes from (i,m) to $(j,m+1)$ if $j\in T_{m+1}$. Also, for each arc (i,j) of length a_{ij} of the original graph, introduce in the expanded graph an arc of length a_{ij} that goes from (i,N) to (j,N).

It is seen that the problem is equivalent to finding a shortest path from $(s,0)$ to (t,N) in the expanded graph.

Let $D^{k+1}(i,m)$, $m=0,1,\ldots,N$, be the shortest distance from (i,m) to the destination (t,N) using k arcs or less. The DP iteration is

$$D^{k+1}(i,m)=\min_{\{j\ |\ (i,j)\text{ is an arc}\}}\left\{\min_{j\notin T_{m+1}}\left\{a_{ij}+D^k(j,m)\right\},\ \min_{j\in T_{m+1}}\left\{a_{ij}+D^k(j,m+1)\right\}\right\},$$
$$m=0,\ldots,N-1,$$

$$D^{k+1}(i,N)=\begin{cases}\min_{\{j\ |\ (i,j)\text{ is an arc}\}}\left\{a_{ij}+D^k(j,N)\right\} & \text{if } i\neq t,\\ 0 & \text{if } i=t.\end{cases}$$

The initial condition is

$$D^0(i,m)=\begin{cases}\infty & \text{if } (i,m)\neq(t,N),\\ 0 & \text{if } (i,m)=(t,N).\end{cases}$$

This algorithm is not very efficient because it requires as many as $N\cdot|I|$ iterations, where $|I|$ is the number of nodes in the original graph. The algorithm can be made more efficient by observing that to calculate $D(i,k)$ for all i, we do not need to know $D(i,k-1),\ldots,D(i,0)$; it is sufficient to know just $D(j,k+1)$ for $j\in T_{k+1}$. Thus, we may calculate first $D(i,N)$ using a standard shortest path computation, then calculate $D(i,N-1)$, then $D(i,N-2)$, etc. This more efficient calculation process may also be viewed as a DP algorithm that involves the solution of N (standard) shortest path problems involving several origins and a single destination. The origins are the nodes in T_k and the destination is an artificial node to which the nodes $j\in T_{k+1}$ are connected with an arc of length $D(j,k+1)$.

Common Problems in Homework 1

1.3:

(1) Some students make an assumption that the machine makes a profit of $100 in the beginning of the period, but this is not reasonable.

(2) Some students compute the optimal cost to go from backward for five weeks (including the first running week). The problem statement actually means in total four weeks, not the beginning week plus four weeks.

1.22

(1) Some students only perform the mathematical induction at the N-2 stages, where the sub-problem has two variables to optimize. This is not complete. A complete solution should include a general case from N-k stage to N-k-1 stage.

2.1

(1) From 1 to 6, it should have two different shortest paths, one is 1->4->6 and the other is 1->2->4->6.

(2) Please write out the shortest path (optimal policy) explicitly.

2.7

(1) Even though this problem carry out a similar proof workflow as the label correcting algorithm in the textbook, in the write up, it should be still written down in details. Some students only include the part of proving termination and existence. These pieces are just the preliminary proof.

2.9

(1) Lots of students misinterpret the problem statement and consider that after the path pass through T_k, it can never go back to T_1, … T_{k-1}. This is not true.

(2) Notice that part b requires a sequence of ordinary shortest path problems involving a single origin and multiple destination, not any type of shortest path problems.

MIT OpenCourseWare
http://ocw.mit.edu

6.231 Dynamic Programming and Stochastic Control
Fall 2015

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.