Case Study: Optimization of Crosscutting and Log Allocation at Weyerhaeuser[5]

Tool: DP

Area of application: Log mill operation

Description of the situation:

Mature trees are harvested and crosscut into logs in different mills to manufacture different end products (such as construction lumber, plywood, wafer boards, and paper). Log specifications (e.g., lengths and end diameters) for each mill depend on the end product the mill produces. With harvested trees measuring up to 100 ft in length, the number of crosscut combinations meeting mill requirements can be large. Different revenues can be realized depending on the way logs are cut from a tree. The objective is to determine the crosscut combination that maximizes the total revenue.

Mathematical model:

The basis of the model is that it is not practical to develop an optimum solution that applies to an "average" tree because, in general, harvested trees come in different lengths and end diameters. Thus optimum crosscutting and log allocation must apply to individual trees.

A simplifying assumption of the model is that the usable length L (feet) of a harvested tree is a multiple of a minimum length K (feet). Additionally, the length of a log cut from the tree is also a multiple of K. This means that logs can only be as small as K feet and as large as NK feet, where, by definition, $N \le \frac{L}{K}$.

Define

$M =$ Number of mills requesting logs

$I = \frac{L}{K}$

$R_m(i, j) =$ Revenue at mill m from a log of length jK cut from the larger end of a stem (or trunk) of length iK, $m = 1, 2, \ldots, M$; $i = 1, 2, \ldots, I$; $j = 1, 2, \ldots, N$; $j \le i$

$c =$ Cost of making a crosscut at point i of the tree, $i = 1, 2, \ldots, I - 1$

$$c_{ij} = \begin{cases} c, \text{ if } j < i \\ 0, \text{ if } j = i \end{cases}$$

The definition of c_{ij} recognizes that if the length iK of the stem equals the desired log length jK, then no cuts are made.

To understand the meaning of the notation $R_m(i, j)$, Figure 12.7 provides a representation of a tree with $I = 8$ and $L = 8K$. The crosscuts at points A and B result in one log for mill 1 and two for mill 2. The cutting starts from the larger end of the tree and produces log 1 for mill 2 by making a crosscut at point A. The cut corresponds to $(i = 8, j = 3)$ and produces the revenue $R_2(8, 3)$. The remaining stem now has a length $5K$. The next crosscut at point B produces log 2 for mill 1 with the length $2K$. This log corresponds to $(i = 5, j = 2)$ and generates the revenue $R_1(5, 2)$. The remaining stem of length $3K$ exactly equals the length of log 3 for mill 2. Hence no further cutting is needed. The associated revenue is $R_1(3, 3)$. The crosscutting cost associated with the solution is $c_{83} = c$, $c_{52} = c$, and $c_{33} = 0$.

The problem can be formulated and solved as a DP model.

[5]Lembersky, M. R., and U. H. Chi, "Decision Simulators Speed Implementation and Improve Operations," *Interfaces*, Vol. 14, No. 4, pp. 1–15, 1984.

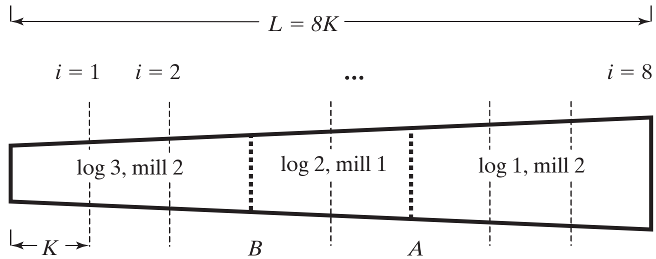


FIGURE 12.7

Typical solution in a two-mill situation

Let

$f(i)$ = Maximum revenue when the length of the remaining stem is iK, $i = 1, 2, \ldots, I$

The DP recursive equation is then given as

$$f(0) \equiv 0$$

$$f(i) = \max_{\substack{j=1,2,\ldots,\min(i,N) \\ m=1,2,\ldots M}} \{R_m(i,j) - c_{ij} + f(i-j)\}, i = 1, 2, \ldots, I$$

The idea is that given a stem of length iK, $f(i)$ is a function of the revenue of cutting a log of length $j\ (\leq i)$ minus the cost of making a crosscut plus the best cumulative revenue from the remaining stem of length $(i - j)K$.

Example computations:

The recursive equation is computed in the order $f(1), f(2), \ldots, f(I)$. The situation deals with two mills $(M = 2)$, a tree of length $L = 12$ ft, and a minimum log length $K = 2$ ft, thus yielding $I = 6$. The cost of a crosscut is $c = \$.15$. Either mill will accept logs of length 2, 4, 6, 8, or 10 ft. This means that $N = 5$. Figure 12.8 provides the spreadsheet solution of the example (file *excelCase8.xls*). The basic DP calculations (rows 15–20) are partially automated and will change automatically when $R_m(i, j)$ in rows 6–11 are altered. All italicized boldface elements are entered manually.[6] The spreadsheet is limited to problems with $I = 6, N = 5$, and $M = 2$, in essence allowing changes in the entries of $R_m(i, j)$ only.[7] The values of $R_m(i, j)$, $j \leq i$, are given in rows 5 through 11 in the spreadsheet. Note that for a specific $j = j^*$, the value of $R_m(i, j^*)$ increases with i to reflect increases in end diameters of the log.

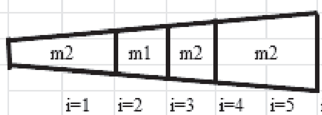
To illustrate the DP calculations in rows 15–20, note that each stage consists of one row because the state of the system at stage i consists of one value only—namely, the partial stem length. At stage $i = 1$, the (remaining) stem length is $1K$, hence resulting in one log only of length $1K$ (i.e., $j = 1$). Also, $c_{11} = 0$ because no cutting takes place. Thus,

$$\begin{aligned} f(1) &= \max\{R_1(1,1) - c_{11} + f(0), R_2(1,1) - c_{11} + f(0)\} \\ &= \max\{1 - 0 + 0, 1.1 - 0 + 0\} \\ &= 1.1 \end{aligned}$$

[6]It is a straightforward Excel exercise to automate columns M and N. I chose not to do that to engage the reader in taking part in determining the optimum solution.

[7]The spreadsheet formulas should provide sufficient information to extend the spreadsheet to other input data. Also, a general spreadsheet solution can be developed using (the more involved) VBA macros to specify the size of the matrices $R_m(i, j)$ and to automate all the calculations.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Input data:													
2	Note: All *italicized* data are supplied manually													
3	m=	*2*	K=	*2*	N=	*5*	L=	*12*	I=	*6*	c=	*0.15*		
4	Rm(i,j):			mill m=1						mill m=2				
5	j-->>	1	2	3	4	5		1	2	3	4	5		
6	1	*1.00*					1	*1.10*						
7	2	*1.10*	*1.15*				2	*1.10*	*2.30*					
8	3	*1.40*	*1.60*	*2.80*			3	*1.33*	*2.40*	*3.40*				
9	4	*1.90*	*1.90*	*3.90*	*4.10*		4	*2.10*	*3.30*	*4.20*	*4.10*			
10	5	*2.10*	*2.90*	*4.40*	*4.80*	*7.20*	5	*2.20*	*3.60*	*4.30*	*4.60*	*6.00*		
11	6	*2.10*	*3.50*	*4.70*	*6.10*	*8.30*	6	*2.20*	*4.50*	*4.40*	*5.00*	*6.30*		
12														
13	Calculations:												f(i)	(j*,m*)
14	i	j=1	j=2	j=3	j=4	j=5		j=1	j=2	j=3	j=4	j=5	*$0.00*	
15	1	1.00						1.10					$ 1.10	*(1,2)*
16	2	2.05	1.15					2.05	2.30				$ 2.30	*(2,2)*
17	3	3.55	2.55	2.80				3.48	3.35	3.40			$ 3.55	*(1,1)*
18	4	5.30	4.05	4.85	4.10			5.50	5.45	5.15	4.10		$ 5.50	*(1,2)*
19	5	7.45	6.30	6.55	5.75	7.20		7.55	7.00	6.45	5.55	6.00	$ 7.55	*(1,2)*
20	6	9.50	8.85	8.10	8.25	9.25		9.60	9.85	7.80	7.15	7.25	$ 9.85	*(2,2)*
21													Value=	$ 9.85



$$f(0) \equiv 0,\ f(i) = \max_{\substack{j=1,2,\ldots,\min(i,N)\\ m=1,2,\ldots,M}} \{R_m(i,j) - c_{ij} + f(i-j)\},\ i = 1,2,\ldots,I$$

FIGURE 12.8

Spreadsheet solution of the mill example problem

The associated optimum decision at $i = 1$ calls for a log of length $1K$ $(j^* = 1)$ for mill 2 $(m^* = 2)$, or $(j^*, m^*) = (1, 2)$.

For stage 2 $(i = 2)$, logs can assume a length of $1K$ or $2K$ (i.e., $j = 1$ or 2) for both mills $(m = 1 \text{ or } 2)$. Thus,

$$\begin{aligned} f(2) &= \max\{R_1(2,1) - c_{21} + f(1), R_1(2,2) - c_{22} + f(0), R_2(2,1) - c_{21} + f(1), \\ &\qquad R_2(2,2) - c_{22} + f(0)\} \\ &= \max\{1.1 - .15 + 1.1, 1.15 - 0 + 0, 1.1 - .15 + 1.1, 2.3 - 0 + 0\} \\ &= \max\{2.05, 1.15, 2.05, 2.3\} = 2.3 \end{aligned}$$

The associated optimum decision is $(j^*, m^*) = (2, 2)$, which calls for cutting one log of length $2K$ for mill 2.

The remaining calculations are carried out in a similar manner as shown in Figure 12.8, rows 15–20. Note that entries B15:F20, H15:L20, and M15:M20 are automated in the spreadsheet. The entries (j^*, m^*) in N15:N20 are created manually after the automated computations in rows 15–20 are completed. Manually highlighted cells in rows 15–20 define $f(i), i = 1, 2, \ldots, 6$.

The optimum solution is read from cells N15:N20 as follows:

$$(i = 6) \rightarrow (j^*, m^*) = (2, 2) \rightarrow (i = 4) \rightarrow (j^*, m^*) = (1, 2) \rightarrow$$
$$(i = 3) \rightarrow (j^*, m^*) = (1, 1) \rightarrow (i = 2) \rightarrow (j^*, m^*) = (2, 2)$$

The solution translates to making cuts at $i = 2, 3$, and 4 and produces a total value of \$9.85 for the tree.

Practical considerations:

The results of the DP optimization model are used by field operators in the day-to-day operation of the mill. Thus the implementation of the model must be user-friendly—meaning that the (intimidating) DP calculations are transparent to the user. This is precisely what Lemberskey and Chi [1] did when they developed the VISION (Video Interactive Stem Inspection and OptimizatioN) computer system. The system is equipped with a database of large representative samples of tree stems from the regions where trees are harvested. The data include the geometry of the stem as well as its quality (e.g., location of knots) and the value (in dollars) for stems with different lengths and diameters. In addition, quality characteristics for the different mills are provided.

A typical user session with VISION includes the following steps:

Step 1: The operator may select a sample stem from the database or create one using the graphic capabilities of VISION. This will result in a realistic representation of the stem on the computer screen. The mills requesting the logs are also selected from the database.

Step 2: After inspecting the stem on the screen, the operator can "cut" the stem into logs based on experience. Next, an optimum DP solution is requested. In both cases, graphic displays of the created logs together with their associated values are projected on the screen. The user is then given the chance to compare the two solutions. In particular, the DP solution is examined to make sure that the created logs meet quality specifications. If not, the user may elect to modify the cuts. In each case, the associated value of the stem is displayed for comparison.

In VISION, DP optimization is transparent totally to the user. In addition, the interactive graphic nature of the output makes the system ideal for training operators and improving their decision-making skills. The design of the system shows how complex mathematical models can be imbedded within a user-friendly computer system.

PROBLEMS

Section	Assigned Problems	Section	Assigned Problems
12.1	12-1 to 12-2	12.3.3	12-23 to 12-27
12.2	12-3 to 12-5	12.3.4	12-28 to 12-30
12.3.1	12-6 to 12-18	12.4	12-31 to 12-32
12.3.2	12-19 to 12-22		