TUGAS OR-2

PENYELESAIAN NETWORK PROBLEM DENGAN MENGGUNAKAN SOLVER DAN LINGO

Rincian Tugas:

1. Kerjakan kembali contoh dan instruksi pengerjaan NETWORK PROBLEM di bawah ini (MAXIMUM FLOW PROBLEM dengan menggunakan SOLVER dan CRITICAL PATH dengan menggunakan LINGO).
2. Cari studi kasus lain, dan selesaikan dengan menggunakan SOLVER dan LINGO.

Catatan:

TUGAS PERORANGAN, kesamaan kasus (no.2) akan mempengaruhi nilai.

Penyelesaian MAXIMUM FLOW PROBLEM dengan menggunakan SOLVER

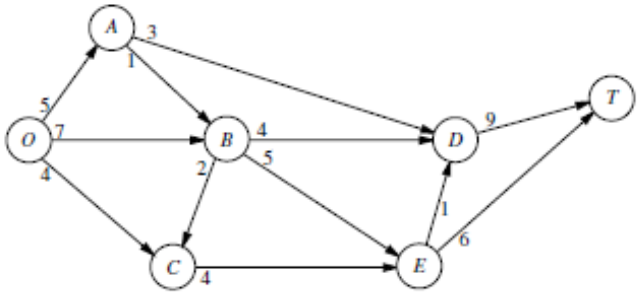


FIGURE 9.6
The Seervada Park maximum flow problem.

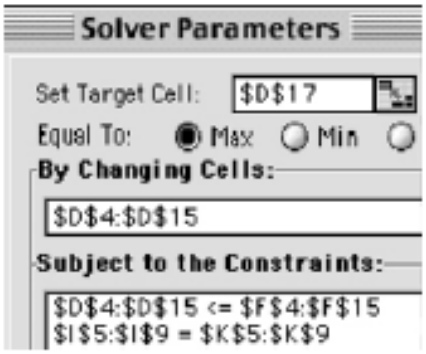
Using Excel to Formulate and Solve Maximum Flow Problems

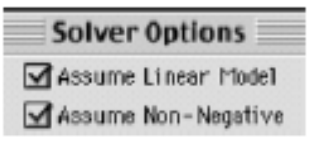
Most maximum flow problems that arise in practice are considerably larger, and occasionally vastly larger, than the Seervada Park problem. Some problems have thousands of nodes and arcs. The augmenting path algorithm just presented is far more efficient than the general simplex method for solving such large problems. However, for problems of modest size, a reasonable and convenient alternative is to use Excel and its Solver based on the general simplex method.

Figure 9.11 shows a spreadsheet formulation for the Seervada Park maximum flow problem. The format is similar to that for the Seervada Park shortest-path problem displayed in Fig. 9.4. The arcs are listed in columns B and C, and the corresponding arc capacities are given in column F. Since the decision variables are the flows through the re-

FIGURE 9.11
A spreadsheet formulation for the Seervada Park maximum flow problem, where the changing cells (D4:D15) show the optimal solution obtained by the Excel Solver and the target cell (D17) gives the resulting maximum flow through the network.

	A	B	C	D	E	F	G	H	I	J	K
1	Seervada Park Maximum Flow Problem										
2											
3		From	To	Flow		Capacity		Nodes	Net Flow		Supply/Demand
4		O	A	4	≤	5		O	14		
5		O	B	7	≤	7		A	0	=	0
6		O	C	3	≤	4		B	0	=	0
7		A	B	1	≤	1		C	0	=	0
8		A	D	3	≤	3		D	0	=	0
9		B	C	0	≤	2		E	0	=	0
10		B	D	4	≤	4		T	-14		
11		B	E	4	≤	5					
12		C	E	3	≤	4					
13		D	T	8	≤	9					
14		E	D	1	≤	1					
15		E	T	6	≤	6					
16											
17		Maximum Flow =		14							





	D
17	=I4

	I
4	=D4+D5+D6
5	=-D4+D7+D8
6	=-D5-D7+D9+D10+D11
7	=-D6-D9+D12
8	=-D8-D10+D13-D14
9	=-D11-D12+D14+D15
10	=-D13-D15

spective arcs, these quantities are entered in the changing cells in column D (cells D4:D15). Employing the equations given in the bottom right-hand corner of the figure, these flows then are used to calculate the net flow generated at each of the nodes (see columns H and I). These net flows are required to be 0 for the transshipment nodes (A, B, C, D, and E), as indicated by the second set of constraints (I5:I9 = K5:K9) in the Solver dialogue box. The first set of constraints (D4:D15 ≤ F4:F15) specifies the arc capacity constraints. The total amount of flow from the source (node *O*) to the sink (node *T*) equals the flow generated at the source (cell I4), so the target cell (D17) is set equal to I4. After specifying *maximization* of the target cell in the Solver dialogue box and then clicking on the Solve button, the optimal solution shown in cells D4:D15 is obtained.

Penyelesaian Identifikasi CRITICAL PATH dengan menggunakan LINGO

EXAMPLE 6 Drawing a Project Network

Widgetco is about to introduce a new product (product 3). One unit of product 3 is produced by assembling 1 unit of product 1 and 1 unit of product 2. Before production begins on either product 1 or 2, raw materials must be purchased and workers must be trained. Before products 1 and 2 can be assembled into product 3, the finished product 2 must be inspected. A list of activities and their predecessors and of the duration of each activity is given in Table 12. Draw a project diagram for this project.

TABLE 12
Duration of Activities and Predecessor Relationships for Widgetco

Activity	Predecessors	Duration (Days)
A = train workers	—	6
B = purchase raw materials	—	9
C = produce product 1	A, B	8
D = produce product 2	A, B	7
E = test product 2	D	10
F = assemble products 1 and 2	C, E	12

Using LINGO to Determine the Critical Path

Many computer packages (such as Microsoft Project) enable the user to determine (among other things!) the critical path(s) and critical activities in a project network. You can always find a critical path and critical activities using LINDO, but LINGO makes it very easy to communicate the necessary information to the computer. The following LINGO program (file Widget1.lng) generates the objective function and constraints needed to find the critical path for the project network of Example 6 via linear programming.

Widget1.lng

```
MODEL:
  1]SETS:
  2]NODES/1..6/:TIME;
  3]ARCS(NODES,NODES)/
  4]1,2   1,3   2,3   3,4   3,5   4,5    5,6/:DUR;
  5]ENDSETS
  6]MIN=TIME(6)-TIME(1);
  7]@FOR(ARCS(I,J):TIME(J)>TIME(I)+DUR(I,J));
  8]DATA:
  9]DUR=9,6,0,7,8,10,12;
 10]ENDDATA
END
```

Line 1 begins the SETS portion of the program. In line 2, we define the six nodes of the project network and associate with each node a time that the events corresponding to

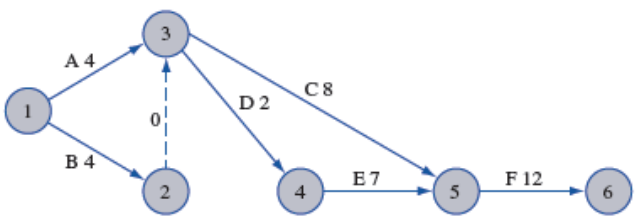


FIGURE 35
Duration of Activities after Crashing

```
MIN     -ET(1 + ET(6
SUBJECT TO
2)- ET(1 + ET(2 >=  9
3)- ET(1 + ET(3 >=  6
4)- ET(2 + ET(3 >=  0
5)- ET(3 + ET(4 >=  7
6)- ET(3 + ET(5 >=  8
7)- ET(4 + ET(5 >=  10
8)- ET(5 + ET(6 >=  12
END

LP OPTIMUM FOUND AT STEP      6
OBJECTIVE VALUE =    38.0000000

                     VARIABLE           VALUE          REDUCED COST
                       ET( 1)        0.0000000E+00      0.0000000E+00
                       ET( 2)         9.000000          0.0000000E+00
                       ET( 3)         9.000000          0.0000000E+00
                       ET( 4)         16.00000          0.0000000E+00
                       ET( 5)         26.00000          0.0000000E+00
                       ET( 6)         38.00000          0.0000000E+00
                    DUR( 1, 2)        9.000000          0.0000000E+00
                    DUR( 1, 3)        6.000000          0.0000000E+00
                    DUR( 2, 3)     0.0000000E+00        0.0000000E+00
                    DUR( 3, 4)        7.000000          0.0000000E+00
                    DUR( 3, 5)        8.000000          0.0000000E+00
                    DUR( 4, 5)        10.00000          0.0000000E+00
                    DUR( 5, 6)        12.00000          0.0000000E+00

                          ROW     SLACK OR SURPLUS     DUAL PRICE
                            1         38.00000          1.000000
                            2      0.0000000E+00       -1.000000
                            3         3.000000       0.0000000E+00
                            4      0.0000000E+00       -1.000000
                            5      0.0000000E+00       -1.000000
                            6         9.000000       0.0000000E+00
                            7      0.0000000E+00       -1.000000
                            8      0.0000000E+00       -1.000000
```

FIGURE 36

the node occurs. For example, TIME(3) represents the time when activities A and B have just been completed. In line 3, we generate the arcs in the project network by listing them (separated by spaces). For example, arc (3, 4) represents activity D. In line 4, we associate a duration (DUR) of each activity with each arc. Line 5 ends the SETS section of the program.

Line 6 specifies the objective, to minimize the time it takes to complete the project. For each arc defined in line 3, line 7 creates a constraint analagous to $x_j \geq x_i + t_{ij}$.

Line 8 begins the DATA section of the program. In line 9, we list the duration of each activity. Line 10 concludes the data entry and the **END** statement concludes the program. The output from this LINGO model is given in Figure 36, where by following the arcs corresponding to constraints having dual prices of -1, we find the critical path to be 1–2–3–4–5–6.

To find the critical path in any network we would begin by listing the nodes, arcs, and activity durations in our program. Then we would modify the objective function created by line 6 to reflect the number of nodes in the network. For example, if there were 10 nodes in the project network, we would change line 6 to **MIN**=TIME(10)–TIME(1); and we would be ready to go!

Widget2.lng

The following LINGO program (file Widget2.lng) enables the user to determine the critical path and total float at each node for Example 6 without using linear programming.

```
MODEL:
  1]MODEL:
  2]SETS:
  3]NODES/1..6/:ET,LT;
  4]ARCS(NODES,NODES)/1,2   1,3   2,3   3,4   3,5   4,5   5,6/:DUR,TFLOAT;
  5]ENDSETS
  6]DATA:
  7]DUR = 9,6,0,7,8,10,12;
```

```
  8]ENDDATA
  9]ET(1)=0;
 10]@FOR(NODES(J) | J#GT#1:
 11]ET(J) = @MAX(ARCS(I,J): ET(I)+DUR(I,J)););
 12]LNODE=@SIZE(NODES);
 13]LT(LNODE) = ET(LNODE);
 14]@FOR(NODES(I) | I#LT#LNODE:
 15]LT(I) = @MIN(ARCS(I,J): LT(J) - DUR(I,J)););
 16]@FOR(ARCS(I,J):TFLOAT(I,J)=LT(J)-ET(I)-DUR(I,J));
END
```

In line 3, we define the nodes of the project network and associate an early event time (ET) and late event time (LT) with each node. We define the arcs of the project network by listing them in line 4. With each arc we associate the duration of the arc's activity and the total float of the activity. In line 7, we input the duration of each activity.

To begin the computation of the ET(J)'s for each node, we set ET(1) = 0 in line 9. In lines 10–11, we compute ET(J) for all other nodes. For J > 1 ET(J) is the maximum value of ET(I) + DUR(I, J) for all (I, J) such that (I, J) is an arc in the network. By using the **@SIZE** function, which returns the number of elements in a set, we identify the finish node in the network in line 12. Thus, line 12 defines node 6 as the last node. In line 13, we set LT(6) = ET(6). Lines 14–15 work backward from node 6 toward node 1 to compute the LT(I)'s. For every node I other than the last node (6), LT(I) is the minimum of LT(J) − DUR (I, J), where the minimum is taken over all (I, J) such that (I, J) is an arc in the project network.

Finally, line 16 computes the total float for each activity (I, J) from total float for activity (I, J) = LT(Node J) − ET(Node I) − Duration (I, J). All activities whose total float equals 0 are critical activities.

After inputting a list of nodes, arcs, and activity durations we can use this program to analyze any project network (without changing any of lines 9–16). It is also easy to write a LINGO program that can be used to crash the network (see Problem 14).