Name: _____

GRAPH OPTIMIZATION MODULE

INTRODUCTION

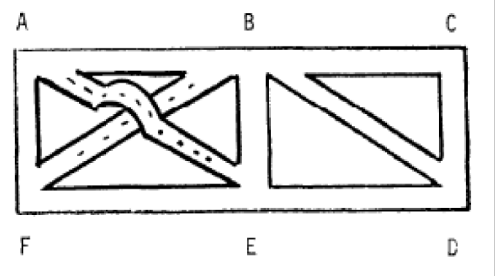
Graph theory has many applications in analysis. Aside from being analogous to physical networks, its elegant form can be adapted to analyze many different problems. A physical application is the analysis of flow patterns of fluids through a network of pipes. But there are many less than obvious applications. In the area of project management, for example, the Program Evaluation and Review Technique and Critical Path Method (PERT/CPM) uses graph theory to best allocate monetary and resources to meet project deadlines. Other applications include laying power or communication lines in the best way, and constructing an "optimal" computer data-base organization. This module will lead you through some basic definitions and some applications of graph theory.

After working through this module, the reader should

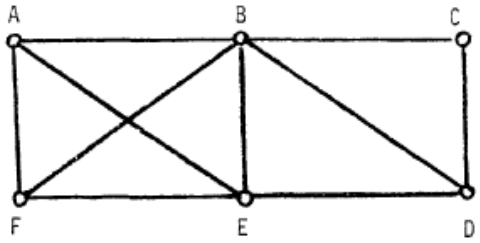
a) have a hands-on feel of how graphs can be manipulated for best performance;
b) see some practical applications of graphs;
c) understand the concept of optimization.

DEFINITIONS

Consider the road map:



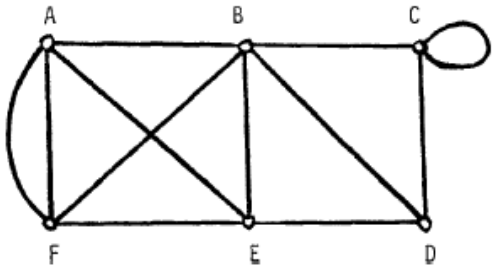
This map can be represented schematically by lines and junction points:



Note that the place where the line AE crosses line BF has no junction point because in

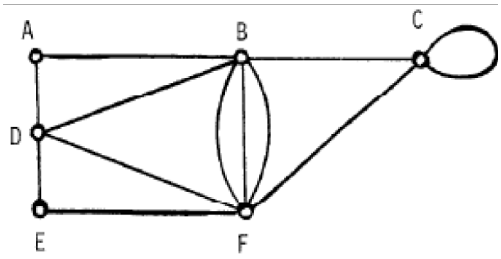
the actual road map, the place is an overpass. The junction points, A, B, C, etc., are called *vertices* (another popular nomenclature is nodes). The lines connecting the vertices are called *arcs* (other popular names include *edges*, *links*, *branches*). The *degree* of a vertex is the number of arcs at that vertex. For example, vertex A of the road map has a degree of three because there are three roads (arcs) at junction A. The diagram with vertices and arcs representing schematically the actual road map is an example of a *graph*. Two vertices can have more than one arc linking the two. A vertex can have an arc from and to itself. Such an arc is called a *loop*.

Consider this graph:



We see immediately vertices A and F are connected by two arcs, and vertex C has a loop,

Illustration 1)

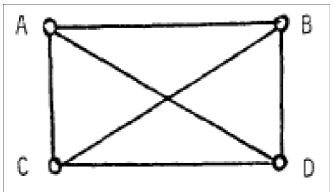


What are the degrees of the vertices? (Please fill in the blank entries in the Table.)

vertex	degree	vertex	degree
A	2	D	
B	6	E	
C	4	F	

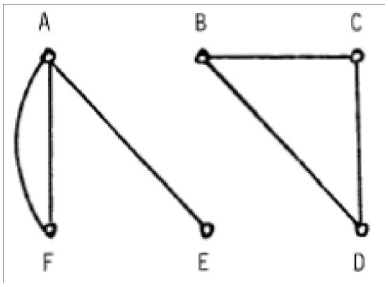
A *path* is a sequence of arcs, one following another from an initial vertex to a final vertex. A *cycle* (or circuit) is a path that starts out and ends at the same vertex.

Illustration 2)



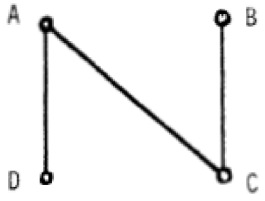
A to B to D to C is a path.
A to B to D to A is a cycle.
A to B to C to D is a ( ).
A to C to D to A is a ( ).
C has a degree of ( ).
B has a degree of ( ).

A *connected graph* is a graph in which any two vertices are connected by a path.
Consider the connection between A to B or E to D in this graph:

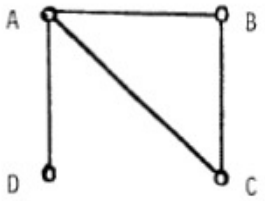


It can be seen that no parts of the graph on the left defined by vertices A, E, F is connected to the right part defined by vertices B, C, and D. This is not a connected graph because vertices B and E are not connected by any path. Notice vertices F and E are connected by the path E to A to F.

A graph in which there is only one path connecting each pair of vertices is called a *tree*. The following is a tree.



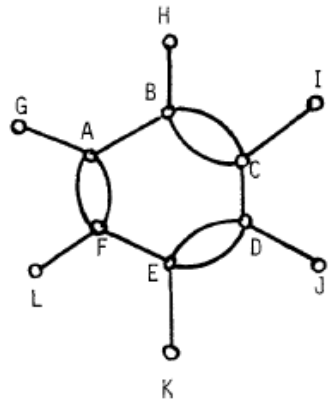
But this is <u>not</u> a tree because there are two paths connecting vertices A and C: A to B to C and A to C.



Here is another way to define a tree: a tree is a *connected graph* which contains no cycle. Can you define a forest?

Illustration 3)

Readers familiar with chemistry may recognize this as the graph for a benzene ring.



What is the degree of each vertex? (Please fill in the empty cells in the Table.)

vertex	degree	vertex	degree
A	4	G	
B	4	H	
C	4	I	
D		J	
E		K	
F		L	

Is it a connected graph?
Is it a tree? If not, what can we change to turn it into a tree? Please show the tree by coloring the appropriate arcs in the given benzene ring graph.

EULERIAN GRAPHS

Illustration 4)

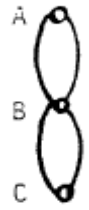
Consider the connected graph:

Can a line be traced over the arcs, starting from a vertex and finishing at the same vertex, with each arc traced only once? The answer is obviously "yes." We start at A, go to B by the left arc, and go back to A by the right arc. (In this module, left and right are from your [the reader's] perspective.)

The degree of A is ( ).
The degree of B is ( ).

Illustration 5)

Consider the connected graph :

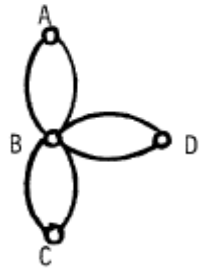


Starting at A, can a closed path (or cycle) be traced over every arc only once? Closed path means the initial and final vertex is the same vertex. We go from A to B on the left arc (the reader's left) and then to C, and then back to B on the right arc, and finally to A.

The degree of A is ( ).
The degree of B is ( ).
The degree of C is ( ).

Illustration 6)

Consider the connected graph:



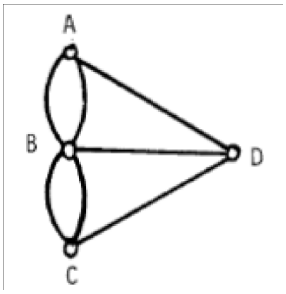
Starting at A, can a closed path be traced over every arc only once?
Yes. On the left side, A to B to C to B (on the right side) to D to B to A.
The degree of A is ( ),
The degree of B is ( ).

The degree of C is ( ).
The degree of D is ( ).

Illustrations 4, 5, 6 are examples of *Eulerian graphs*. An Eulerian graph is a graph on which an Eulerian path can be traced. An Eulerian path starts out at a vertex, traces out each arc once, and ends at the starting vertex.

Rule (1)

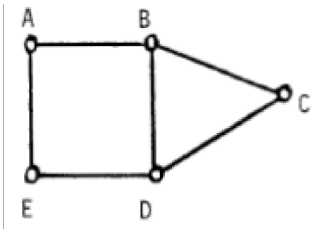
A connected graph is Eulerian if and only if the degree of every vertex of the graph is even. The graphs in Illustrations 4, 5, 6 are *Eulerian* because every vertex of the graph is even. The following famous Konigsberg bridge example is not an Eulerian graph because vertices A, B, C and D have degrees of odd number.



Rule (2):

A connected graph is *semi-Eulerian* if only two vertices have degrees of odd number. Semi-Eulerian means that if we start out at an initial vertex, and trace out every arc once, we end at a different final vertex.

Illustration 7)



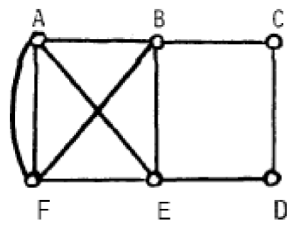
Please fill in the following Table:

Vertex	Degree
A	2

Is the graph semi-Eulerian and why?

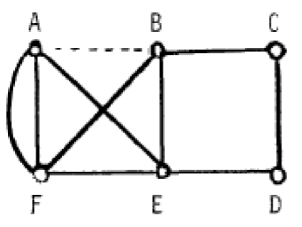
Illustration 8)

Consider an Eulerian graph;



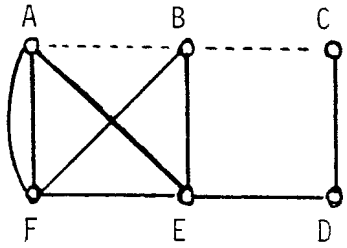
It is an Eulerian graph because every vertex has a degree of even numbers, We trace out an Eulerian graph to gain some insights of the algorithm known variously as Euler's algorithm and Fleury's algorithm.

a) We go from A to B and erase the traversed arc:



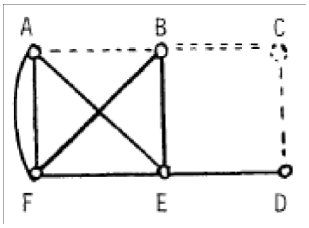
Vertices C, D, F have even degree. Vertices A and B have odd degree. The graph now has two vertices with degree of odd number--it is a semi-Eulerian graph. (See Illustration 7.)

b) We go from B to C and erase the traversed arc:



Vertices B, D, E, F are even. Vertices A, C are odd. It is a semi-Eulerian graph.

c) We go from C to D and erase the traversed arc:

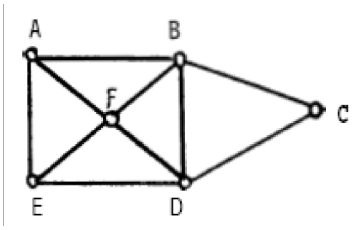


Vertices B, E, F are even. Vertices A, D are odd. It is a semi-Eulerian graph.

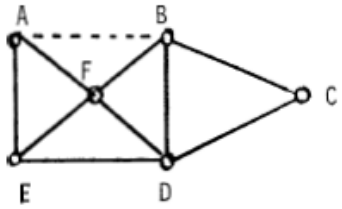
From these three examples, we conclude that when we traverse on an Eulerian graph from one vertex to another vertex, in this case A to B, and we erase the traversed arc, the two vertices, A and B, become odd degree vertices. Convince yourself that after traversing each arc, there remains two vertices with odd degree by completing this illustration. (Please show three examples.)

Illustration 9)

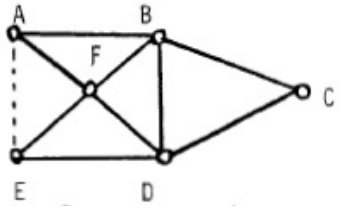
Instead an Eulerian graph, suppose we have a semi-Eulerian graph:



It is a semi-Eulerian graph because vertices A and E are odd degree vertices.
We go from A to B and erase the traversed arc:



Vertices A, C, D, F are even degree. Vertices B and E are odd degree. Work out this example to convince yourself that always two vertices with odd degree or all vertices are even degree, We can visualize the latter case by going initially from A to E instead of A to B:



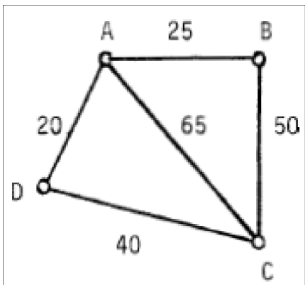
Now please show three additional examples (at the minimum).

APPLICATIONS

Graph theory is not just drawing lines and promenading across bridges. The textbook discusses the "traveling salesman problem," defined as the least costly tour for a salesman to visit all the required stops. In this section we look at another application: the minimum spanning tree. Go back to the beginning of the module if you are not clear on the definition of a tree.

Illustration 10)

Suppose we have four buildings connected by utility tunnels. The numbers represent distances in meters. We want to connect each building to the others with a power line through the utility tunnels. We also want to use the minimum length of power line to save money.



The algorithm:

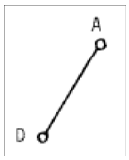
a) List the arcs according to increasing magnitude:

AD	AB	CD	BC	AC
20	25	40	50	65

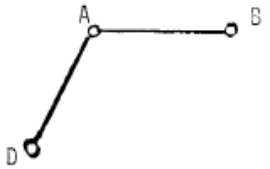
b) Select arcs, beginning from the top of the list, until we have a tree for the graph. (This means we skip an arc on the list to avoid loops or cycles.) The sum of the selected arcs is the minimum length.

Procedural steps:

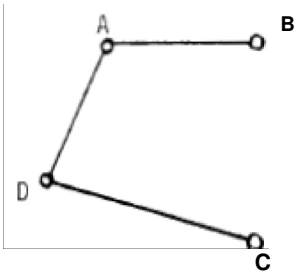
1) We pick AD



2) We pick AB



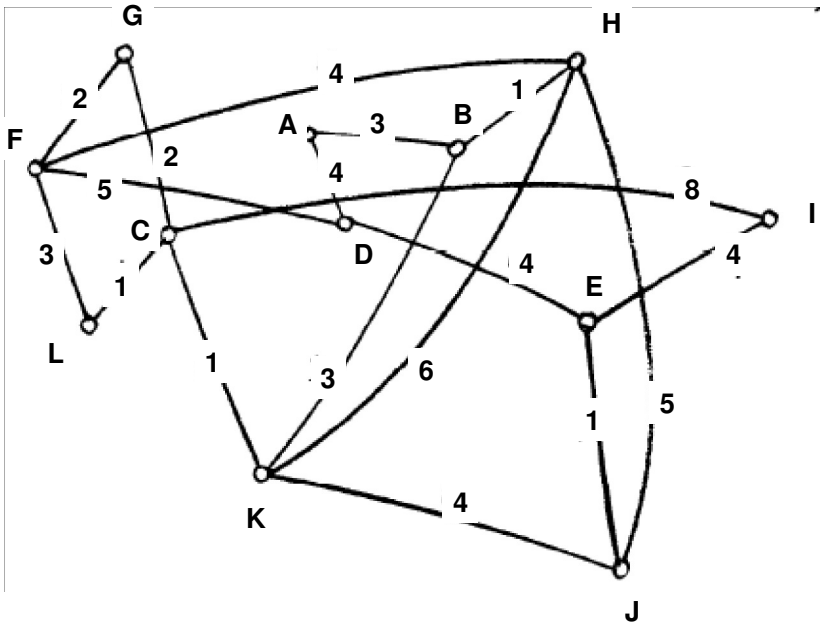
3) We pick CD



When step 3 is reached, the graph is a tree—a minimum spanning tree because the total path length is the minimum of all possible spanning trees of this graph. For this case it is 85 meters.

Illustration 11)

Shown below are the passenger terminals and routes (the distances are in kilometers) in a computerized transportation system. The control box at each terminal must be connected to all the other terminals through a cable. Along which routes should the cable be laid to achieve a minimum spanning tree? What is the total length of the cable? The arcs are not in scale proportion. (Please show the route in the graph below with a color pen or pencil. Notice cables can be spliced together. They do not need to form a continuous piece. Then complete the accompanying Table.)



Arc	Length	Arc	Length
LC	1	KJ	
CK	1	DE	
JE	1	IE	
BH	1		
FG	2		
CG			
LF			
KB			
AB			

Total length of minimum spanning tree is ( ).

Illustration 12)

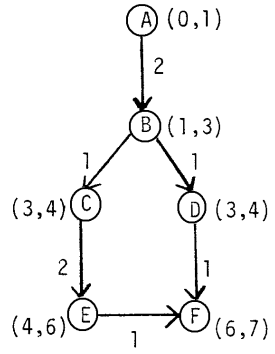
The techniques for large scale project management using graph theory began development in the late 1950's. Large projects involve numerous (in the hundreds and thousands) interrelated activities. The objective of careful planning, scheduling, and coordinating these activities a bewildering goal without the aid of quantitative techniques. PERT/CPM was one of the techniques developed to aid management. In the original versions, there are technically differences between PERT (Program Evaluation and Review Technique) and CPM (Critical Path Method) In recent applications, however, both have merged and the analysis is simply named PERT-type system. In this section, we will study the basic definitions of PERT analysis, work out one simple example, and at the end, you will work on a more complex example for exercise.

Suppose we are building a house. We identify six basic activities:

Activity	Depends on the completion of	Estimated months	Personnel
A, foundation		1	4
B, rough wall	A	2	4
C, utilities	B	1	3
D, roof	B	1	4
E, wall board	C	2	2

F, interior and exterior painting and finishing	D, E	1	3

PERT-type system analysis is based on *precedence* relationships. For example, activity B, rough wall, cannot begin until activity A, foundation, is completed; activity F cannot begin until activities D and E are completed. The six basic activities can be represented by a graph:



Each vertex (or node) represents the *completion* of an activity. The *directed arc* (or edges, branches) represent precedence relationships. Each arc also represents the duration, in time, of the *next activity*, i.e., the vertex to which the arc is pointing. For example, the two arcs emanating from vertex B are numbered 1 because both activities C and D take one month to complete.

We can also identify earliest start time and earliest finish time for this project:

Activity	Earliest start time	Earliest finish time
A	0 + 1 (duration)	= 1
B	1 + 2 (duration)	= 3
C	3 + 1 (duration)	= 4
D	3 + 1 (duration)	= 4
E	4 + 2 (duration)	= 6
F	6 + 1 (duration)	= 7

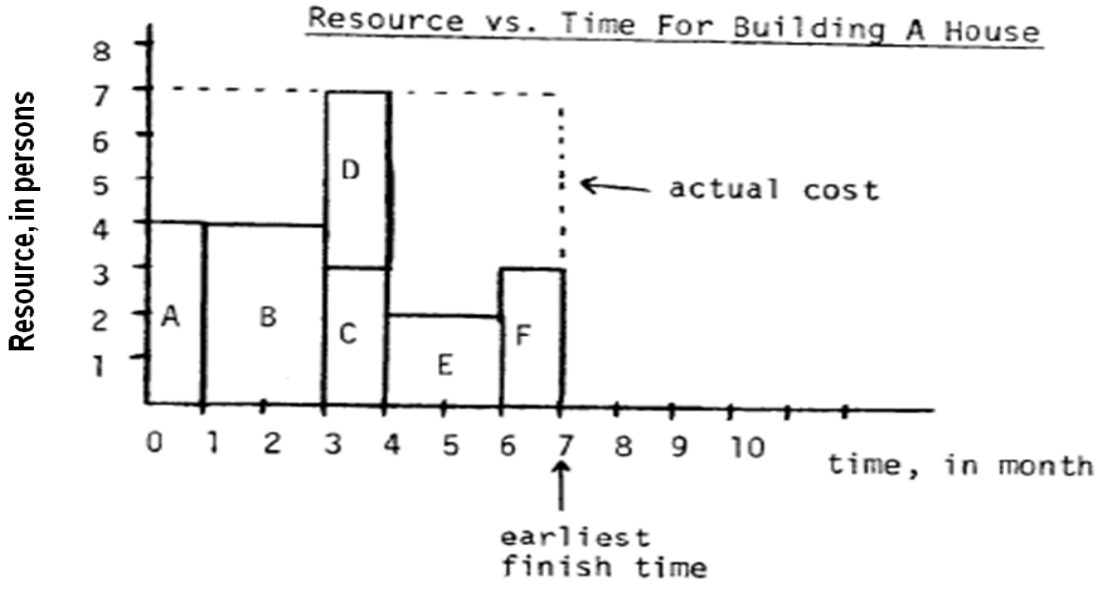
Earliest start time is defined as the time which an activity can begin if all preceding activities are started as early as possible; obviously, finish time of an activity is the earliest completion time for the activity if the activity is started at the earliest time. We put (S , F) next to the vertices to indicate the earliest start (S) and finish (F) times. For example, (0, 1) next to A means an activity the initial activity A will start on at the initial time (0th month), and if everything goes well, activity A will finish at the end of the 1st month. The (1, 3) next to B means activity B can begin at the *earliest start time* at the

end of the first month. Because activity A will take one month to complete, therefore its earliest finish time is 1. Activity B takes two months to complete, therefore, its earliest finish time = 1 + 2 = 3.

Activity F cannot begin until both activities D and E are completed. Though the earliest finish time for activity D is 4, the earliest finish time for activity E is 6; thus the earliest start time for activity F is 6, not 4. In PERT analysis language we say activity D has a slack of 2. It has a slack of 2 because D's earliest finish time is 4 and the next activity F cannot begin until the 6th month. Also in PERT analysis language, we say the path, B to C to E to F is the critical path. It is the critical path because it has the longest total path length and any delay in the activities represented by this path will cause the delay of the completion of the project. Path B to D to F is not the critical path because it has a *slack* of 2 months, i.e., the activities on this path can be delayed 2 months without causing the delay of the completion of the project if a11 other activities are on time. From this brief discussion we arrive at one of the objectives of PERT analysis—to identify the activities that are most likely to be bottlenecks. For this example activities B, C, E are potential *bottlenecks* because they are on the critical path.

Notice there may be more than one critical path in a project. For example, if the link from E to F is missing in the graph on Page 11, then there are "two products" for this project; one is the wall board and the other is the painting and finishing. In this case, there are two critical paths. One is A-B-C-E and the other is A-B-D-F. Obviously, the numbers contained in the parenthesis next to F have to be changed now to read (4, 5), since it takes 6 units of time before the wall board is completed, and it takes 5 time units to complete the painting. The only situation we have a single critical path is when the graph was drawn in the original manner, when there is only one node, F, as the terminal node, representing there is only one final product in the entire project.

We consider another objective of PERT analysis: to evaluate the effect of changes, such as a shift of resources, on the project. Consider the chart:



The chart is a plot of resource required (in person-months) versus time (in months). For example, activity A requires 4 persons and the duration is 1 month; activity B requires 4 persons with earliest start time at the 1st month and earliest finish time at the 3rd month, and so on. The *cost* of this project measured by labor is the total area of the 6 blocks. For example, if the cost of one man-month is $1,000. the cost of activity A is

4 persons x 1 month x $1,000/person-month = $4,000.

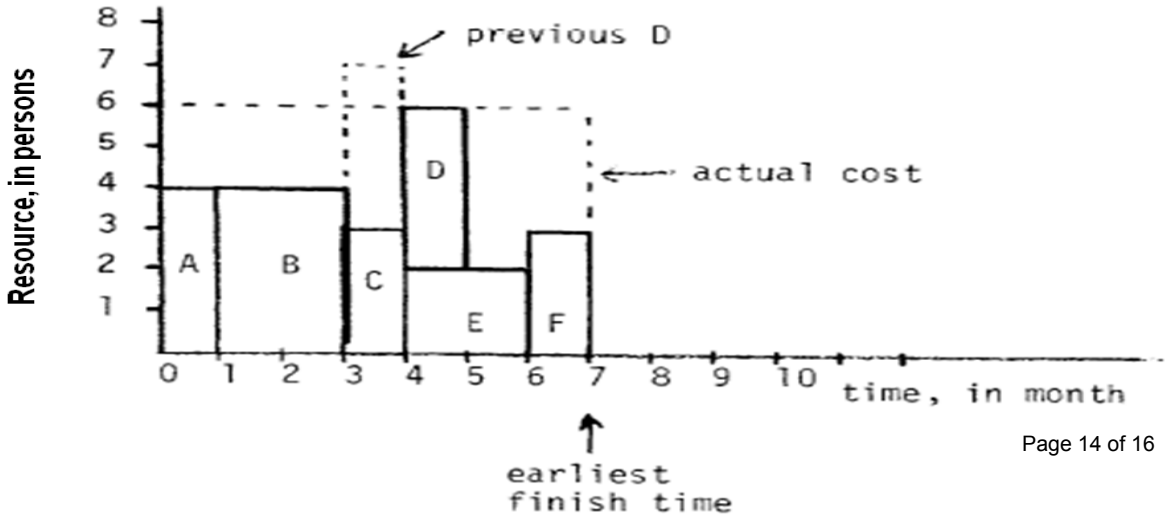
(Please complete the following Table.)

Activity	Cost
A	$4,000
B	$8,000
C	
D	
E	
F	
TOTAL	$26,000

The total labor cost is $26,000. This is called *intrinsic cost*. The actual cost is

7 persons x 7 month x $1,000 = $49,000.

The total labor cost is $26,000. The actual cost is 7 persons X 7 month X $1,000 = $49,000. We get this number because when activities C and D are simultaneously performed, we need 7 persons. It means we have to have 7 persons on the payroll for the duration of the project. On the chart, the area represented by the dashed line is the actual cost. A method to reduce the actual cost is to use a technique known as resource leveling. Activity D is not on the critical path ("off critical path" in PERT vernacular). Activity D has a slack of two months, and we can move it as shown below:



With this shift of resource, the actual cost is

6 persons x 7 months x $1 = $42,000,

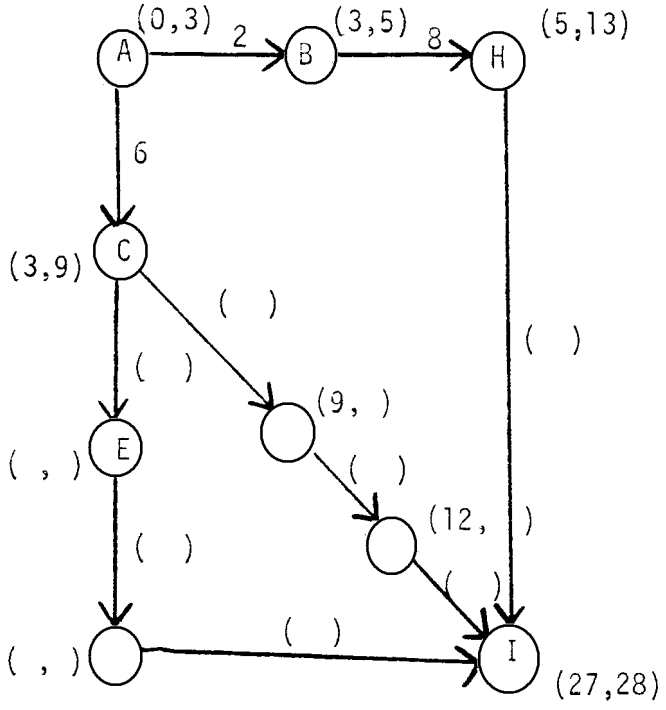
a decrease of $7,000 in actual cost.

Illustration 13)

Activity	Depending on completion of	Estimated months	Personnel (in persons)
A		3	2
B	A	2	3
C	A	6	2
D	E	8	2
E	C	10	1
F	C	3	1
G	F	3	1
H	B	8	1
I	D, H, G	1	1

Given this project task outline and labor requirements, do a PERT analysis. What is the earliest finish time of the project? What is the slack in activities H? Assume project cost at $2,000 per person-month, and do a resource leveling to achieve minimum labor cost. Do the resource leveling in the graph on page 16.

As a first step, please complete this graph:



The critical path is A to ( ) to ( ) to ( ) to I.

Now fill in the missing blocks (D, F, G, H, I) for the resource vs. time chart. Then answer the following questions:

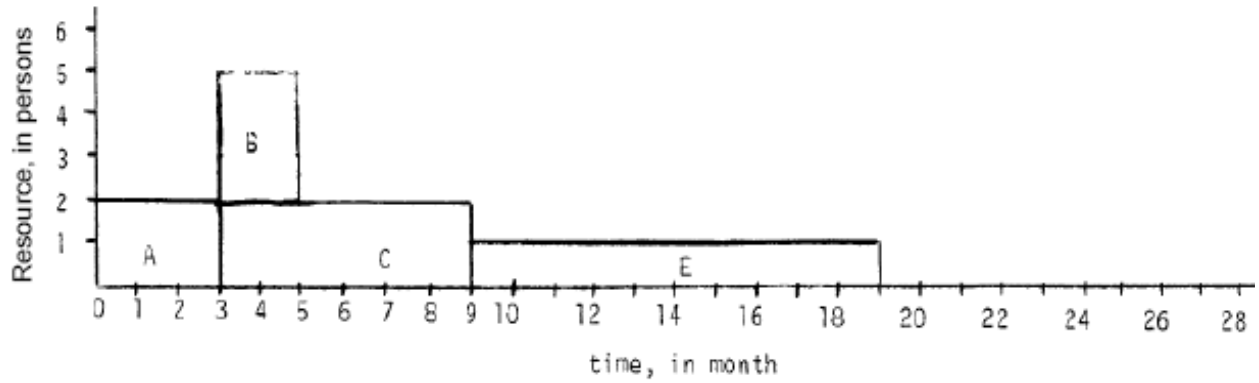
The intrinsic labor cost is ( ).

The actual labor cost is ( ) before leveling.

Slacks:

Earliest start for D is 19.
Latest start time for D is 19. Slack for D = 0

Earliest start time for F is 9.
Latest start time for F is ( ). Slack for F =



Earliest start time for H is ( ).
Latest start time for H is ( ). Slack for H = ( )

Finally, do a resource leveling for this project to minimize labor cost, with the same completion time of 28 months. What is the total person-month? What is the minimum actual labor cost? (Draw the resource leveling graph below. Notice the sequence of activities remains unchanged when you perform resource leveling.)