The sequencing of several project networks on limited facilities is discussed under the assumption that the projects and resources have already been specified by a higher scheduling function.

A priority function is proposed which uses both the local and global properties of the project network.

The resulting schedule is then converted into a network on which useful alterations can be made.

Fabrication and assembly operations

Part V Production order sequencing

by A. B. Calica

Between conventional techniques of scheduling and the actual production order sequencing lies a gap which should be bridged to provide a logical system connection between the planning process and the implementation of the shop loading function. Project network techniques, including PERT, generally do not consider the constraints on resources.[1,2] It is true that resource utilization can be extracted from a PERT-type schedule. However, there is no guarantee that the amount of resources required for any scheduled interval of time will not exceed the plant capacity, thereby making the schedule invalid. Also, if the demand for resources associated with the PERT schedule fluctuates severely, the resulting schedule may be expensive. These considerations are especially acute when several project networks compete for limited resources. On the other hand, by definition, sequencing techniques provide a feasible loading of the shop.[3] However, this loading is usually accomplished at the hazard of ignoring project goals in favor of local (short-term) increases in facility utilization. An exception is discussed in a report by B. Banerjee,[4] which is a treatment of a set of project-oriented heuristics for shop sequencing.

In this discussion, we describe a priority function for sequencing that takes into account global as well as local properties of the

project networks. Also, the result of such sequencing is shown to be expressible as a PERT-type network, called the *combined network*, whose structure includes the resource constraints of the plant. The combined network provides information for the analysis of a schedule and suggests certain techniques for altering the schedule without having to resequence the activities.

Presently, optimality in solving most industrial sequencing problems is not economically feasible. The basic concepts of reasonable scheduling procedures for cases in which human intervention is impractical are developed in this paper.

Throughout this discussion, the term *project* denotes a network of activities leading to the production of a definable end item (which could be a single unit or a multiplicity of units). It is assumed that no project contains any activities in common with any other project. An *activity* is a set of one or more elementary processing steps. An activity might correspond, for example, to a single machining operation using a single resource.

sequencing

The subject considered here is the local multiproject, multifacility scheduling problem. For purposes of this description, the following environment is assumed: (1) the status of the shop at time 0 is known, (2) the length of the scheduling period and a due date have been specified for each project, (3) each project has a network representation, (4) the resources for any scheduling period are known, although not necessarily fixed, and, (5) an activity, once started on a machine, will not be interrupted. Under these restrictions, control can be exercised through the use of the following techniques:

- Sequencing
- Assignment of additional resources (e.g., overtime)
- Lot splitting

The problem of scheduling with sequencing as the sole control technique is now examined. A feasible schedule can be imagined as consisting of all the decisions that load the available activities on available machines. For each point in time at which a decision is to be made, a priority function f is assigned to each activity competing for an available resource. Lower values of f indicate higher priority. Thus, the activity with the smallest value of f has the available resource assigned to it.

Suppose that activity A_1 of project P_1 and activity A_2 of project P_2 are both competing for the use of a single machine at time t. Then $f(A_1, P_1, t)$ and $f(A_2, P_2, t)$ are computed, and activity A_1 is loaded if $f(A_1, P_1, t) < f(A_2, P_2, t)$. The priority function f should reflect the current status of the system, induce decisions compatible with the overall objectives of management, and be easily computable.

For an activity A of project P, ready to be loaded at time t, we introduce the following variables associated with this activity and this project:

M expected duration of activity A

F total float (latest start time minus earliest start time minus M) of activity A at time t

S project slack (difference between due date and earliest possible completion date) at time t

D node density (as defined later).

We now define a priority function f with arguments M, F, S, and D. Without specifying this function, we can state the general properties required of it: f increases with M, F, and S, and decreases with D. The priority increases with decreasing f, and consequently increases with decreasing M, F, and S, and with increasing D. Heuristically, this means:

$$\frac{\partial f}{\partial M} \geq 0, \qquad \frac{\partial f}{\partial F} \geq 0, \qquad \frac{\partial f}{\partial S} \geq 0, \qquad \frac{\partial f}{\partial D} \leq 0.$$

Smaller values of M tend to give an activity higher priority, thus tending to shorten the average waiting time for activities in queue and reduce the average number of activities in queue. Ordering by the use of M alone is known as the *shortest operation discipline.*[5]

Smaller values of F are indications of the critical nature of the activity with respect to the rest of the project. If F is zero, for example, this indicates that the activity is critical in the sense of CPM terminology,[6,7] and that a delay in starting will cause a delay in the currently expected minimum completion date of the project. Conversely, larger values of F would indicate that the activity could be delayed up to F units of time without affecting the expected minimum completion date of the project. Note that, by definition, F is always greater than or equal to 0.

The project slack, S, is the amount of time that the completion date may be increased without exceeding the target date for the completion of the project. A negative value for the slack indicates that the project will be late by at least the value of S. Low values of S are a rough indication of project lateness and strongly suggest high priority.

The node density, D, is a global network property, high values of which are indicative of branching later in the project network, and hence merit higher priority.

The values of M and D are assumed time invariant and can thus be stored and computed before the activities are actually sequenced. The values of F and S, on the other hand, vary as a function of time. Therefore, a sequencing rule using F and S is at least to some extent self-corrective.

The class of functions f that satisfies the conditions set forth is quite large. A class of functions recommended for consideration is of the general form:

$$f = \frac{a_1 M^{b_1} + a_2 F^{b_2} + a_3 S^{b_3}}{D^{\alpha}},$$

where α, a_1, a_2, a_3, b_1, b_2, and b_3 are positive constants or functions of the independent variables. If ease of computation is considered, this form can be further restricted to the special case:

$$f = \frac{a_1 M + a_2 F + a_3 S}{D^{\alpha}}. \tag{1}$$

The Appendix presents a specific example of the form:

$$f = \begin{cases} \dfrac{M + F + S}{D} & S \geq 0 \\ \dfrac{M + F}{D} & S \leq 0. \end{cases}$$

There is reason to believe that a function of form (1) would be sufficient for most loading purposes. The choices of a_1, a_2, a_3, and α can be a reflection of management decisions concerning the weighting of various factors in the operation, such as in-process inventory, lateness, and resource utilization. These choices may be augmented by the status of the activities in the shops, as explained in Part VI of this paper. For example, if a numerical index (e.g., cost) is assigned to a schedule operating in a given period, this index can be used as a basis for mapping a_1, a_2, a_3, and α and adjusting these constants in an adaptive fashion.

node density

It is clear that a local priority function for the assignment of priorities in project networks should take into account some of the topological properties of the network under consideration. For this reason, the function D_i, the *node density*, is now defined, denoting n_i as the ith node of the network under consideration.

$$D_i = \frac{\text{Total number of estimated hours on all branches following } n_i}{\text{Minimal estimated time necessary to complete the project } (n_i \text{ to end of project})}$$

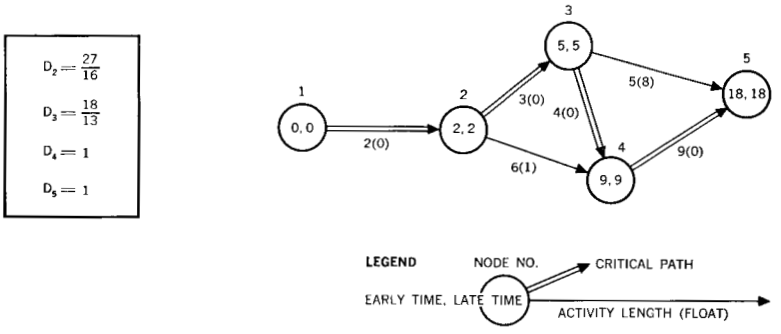
or 1 if not otherwise definable.

This calculation can be illustrated by the example shown in Figure 1.

The use of D_i as a variable in a priority function f, where $\delta f / \delta D_i < 0$, has the qualitative effect of tending to give the higher priority (lower values of f) to activities that are predecessors of highly parallel processing.

The above definition of D_i has the advantage of being intrinsic to the original formulation of the network and does not have to be recomputed during the sequencing of the project. Although a value of D_i larger than 1 indicates that parallel processing must be expected, an individual value of D_i is not sufficient to indicate where this parallel processing will occur. It might be helpful to define a vector function at each node, which could then be multiplied (inner product) by an appropriate state vector to yield an index reflective of the system condition when node i is reached.

Figure 1 Example for calculating D_i



network combination algorithm

Once the activities have been sequenced by the priority function, the resulting schedule can be brought into network form by use of a network combination algorithm. This is an algorithm for amalgamating the project networks of one or more projects into a single network having the following properties:

- The constraints imposed by the sequencing rule and resource availability appear as precedence relations in the combined network.
- The combined network is a directed, acyclic PERT-type network that contains a single source and a single sink.

The combined network contains the information of the original project networks, the sequencing rule, and the resource restrictions, all of which are contained in a Gantt Chart[8] of the final schedule. PERT-type calculations can be performed on the combined network, which demonstrates in a single configuration a continuous class of feasible schedules of which the Gantt Chart only expresses the node time and earliest-event time.

Constructing a network that contains all the above properties might sound trivial, since one can combine the original project networks into a single network by merely inserting arrows between activities in the original networks to indicate the precedences resulting from sequencing decisions. Unfortunately, the network obtained in this manner is not necessarily acyclic. If an acyclic network is desired, a more meticulous procedure must be followed. A network having the specified properties is obtained from the following 3-step construction:

Step 1. Arrange the activities in linear chains corresponding to their ordering function for each facility. Insert an arc of length zero between the end of one activity and the beginning of its successor in the linear chain that has been formed.

Step 2. If activity A_1 directly precedes activity A_2 in one of the original PERT networks, and A_1 and A_2 are executed on different facilities, insert in the combined network an arc of length zero between the node directly succeeding A_1 and the node directly preceding A_2.

Step 3. Identify all initial nodes to create a source for the network. Identify all terminal nodes to create a sink for the network.

For example, suppose that there are two projects which use the three facilities A, B, and C. The projects are represented as shown in Figure 2, where an activity is expressed by two numbers in parentheses denoting the predecessor and successor nodes of this activity. Using the loading rule $f = F + M$, the sequences of Table 1 are produced at each facility. Application of Steps 1 and 2 results in the structure of Figure 3. Identifying the initial and terminal nodes, we arrive at the PERT network of Figure 4 upon which the usual PERT/CPM calculations are performed. The critical path of the combined network does not necessarily correspond to the set of critical activities in the original networks.

Theorem: The combined network is acyclic.

Proof: If the node times in the combined network are the actual early-event times, as is the case on the Gantt Chart of the Appendix, all arcs in the network either move forward in time or stay even. No arc moves backward. An arc that moves forward in time cannot be part of a cycle, because such a cycle would require an arc that moves backward in time in order to close the cycle. If we were trying to form a cycle solely with arcs that do not move forward but stay even, such a cycle would contain at least one node that has an even-staying arc directed into it and also an even-staying arc directed out of it. However, such nodes do not exist because (1) nodes at the beginning of a production activity can have even-staying arcs directed only into them but not out of them, (2) nodes at the end of a production activity can have even-staying arcs directed only out of them but not into them, and (3) we can divide all nodes into two classes that have no elements in common: those at the beginning of a production activity and those at the end of a production activity. Therefore, no cycle can contain arcs that move forward in time or stay even. QED

To find the size of the combined network, suppose that there are k projects $(P_1, \cdots, P_k)$ and m machines $(J_1, \cdots, J_m)$ such that c_i is the number of arcs in project P_i $(i = 1, \cdots, k)$, $\tilde{c}_i$ is the number of arcs from all projects for machine J_i $(j = 1, \cdots, m)$, and A^* is the number of activities in the entire program. Then

$$A^* = \sum_{i=1}^{k} c_i = \sum_{j=1}^{m} \tilde{c}_j.$$

Let

$$R_i = \sum_{n} \rho_n \rho_n^*,$$

where the domain of n is the set of nodes of project P_i, where ρ_n

Table 1 Activity sequences

Facilities	*Activity sequences*			
A	(1, 2)	(3, 4)	(6, 8)	(7, 8)
B	(1, 3)	(7, 9)	(4, 5)	(8, 9)
C	(6, 7)	(2, 4)		

Figure 2 Project representation

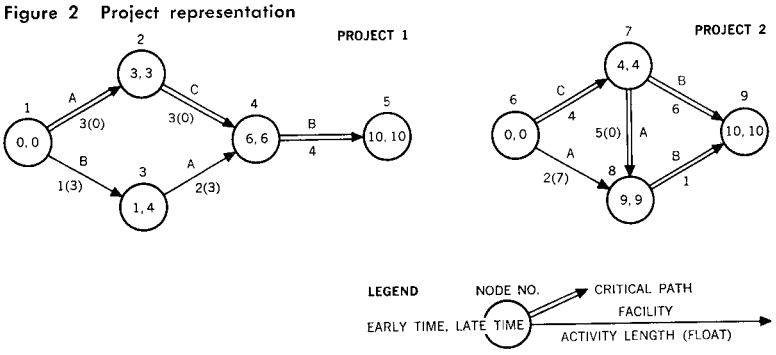


Figure 3 Network construction

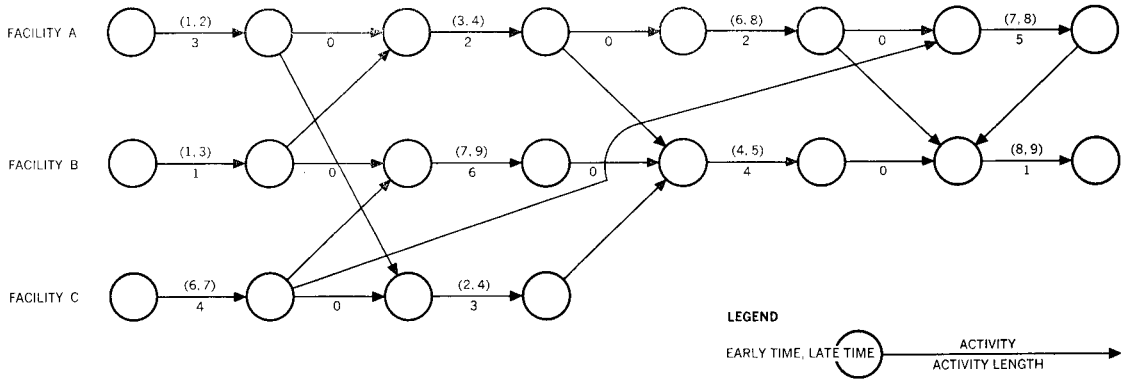
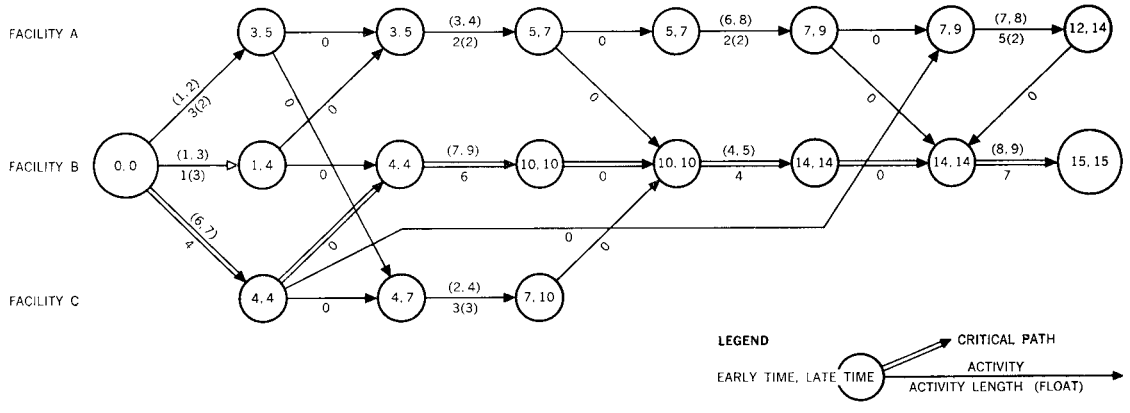


Figure 4 PERT network



denotes the number of arcs directed into node n, and where ρ_n^* denotes the number of arcs directed out of node n.

If N denotes the number of nodes in the combined network, then $N \leq 2A^*$. If C denotes the number of arcs in the combined network, then

$$C \leq (2A^* - m) + \sum_{i=1}^{k} R_i,$$

where

$$R_i \leq \frac{c_i^2 - c_i}{2}.$$

However, by replacing nodes by dummy arcs, where

$$\rho_n \rho_n^* > 1 + \max(\rho_n, \rho_n^*),$$

networks can be arranged in such a manner that $R_i \approx c_i$. Therefore, as a rough estimate, we can say that $C \leq 3A^*$. In particular, networks associated with pure assembly processes have $C \leq 3A^*$. It is possible to eliminate some of the unnecessary zero-length arcs, but this must be done systematically to avoid creating cycles in the combined networks.

The combined network can be used to identify the critical path in the system, and can serve as input to a project compression algorithm if the method proposed by Ford and Fulkerson,[9] or an appropriate variant, is used.

Within the combined network, activities can be delayed up to their float without affecting the completion date of the final node. Lot splitting can be done if no server is held up past the late start date of an activity in the combined network, again guaranteeing the completion date of the sink node of the combined network. Finally, subnetworks of the combined network can be meaningfully extracted and compressed so that the sink node of the subnetwork is advanced toward the present time without adversely affecting the completion date of the entire combined network.

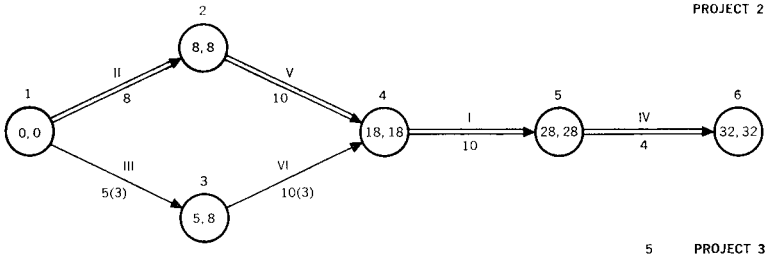
compressed network

Assuming that additional resources can be added to individual servers in the system to shorten some of the activities, it is then possible to assign these extra resources to the combined network in such a way as to compress the entire combined network. This compression technique produces a schedule that has a combined network with changed arc and node times. This compressed network is structually equivalent to the original network.

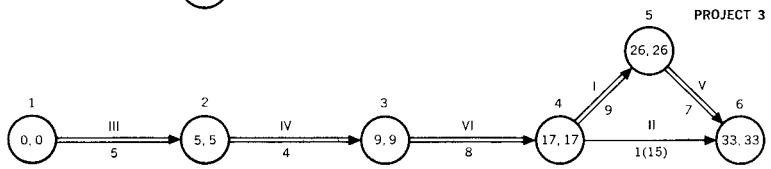
The combined network can be used for the following purposes:

- Determination of the critical path through the schedule
- Dynamic lot-splitting decisions
- Selection of machines for tasks not specified in the original plans
- Alternate routing decisions
- Input to a project compression algorithm
- Determination of admissible start time delay (total float) for each task

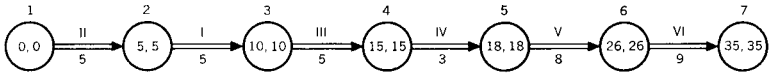
PROJECT 2



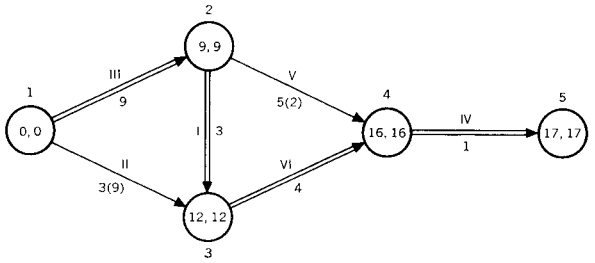
PROJECT 3



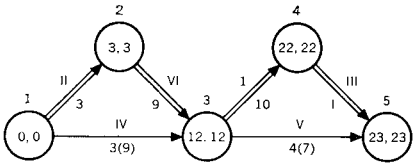
PROJECT 4

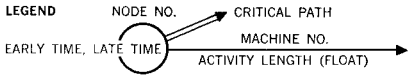


PROJECT 5



PROJECT 6





Program duration:

a priori duration $\geqq 50$

actual duration $= 56$

Summary of time requirements

PROJECT	MACHINE I	II	III	IV	V	VI	TIME/PROJECT
1	3	6	1	3	6	7	26
2	10	8	5	4	10	10	47
3	9	1	5	4	7	8	34
4	5	5	5	3	8	9	35
5	3	3	9	1	5	4	25
6	10	3	1	3	4	9	30
	40	26	26	18	40	47	197
	TIME/MACHINE						Total time

Node densities

PROJECT	NODE 1	2	3	4	5	6	7
1	1.37	1.39	1.47	1.00	1.00	1.00	—
2	1.47	1.00	1.00	1.00	1.00	1.00	—
3	1.03	1.04	1.04	1.06	1.00	1.00	1.00
4	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5	1.47	1.63	1.00	1.00	1.00	—	—
6	1.31	1.20	1.36	1.00	1.00	—	—

Symbols and abbreviations used in this example

- P Project number
- J Machine number
- M Processing time
- t Clock time
- Δt $t_i - t_{i-1}$
- S Project slack at t_i
- F Activity float at t_i
- D Node density
- f Value of priority function
- L This activity starts (is loaded) at t_i
- A This activity is brought into queue (added) at t_i
- R This activity is running
- O This activity is taken off at t_{i+1}

Initial slack computations ($t_0 = 0$)

PROJECT	SLACK
1	31
2	18
3	17
4	15
5	33
6	27

Detailed sequencing

TIME	JOBS ON OR IN QUEUE AT t_i								JOBS ON AT t_i				SUMMARY
t_i	P	J	M	S	F	D	f		J	P	M		
$t_0 = 0$	1	III	1	31	0	1.39	23.0		I	—	—		Jobs off this step: 0
	2	II	8	18	0	1.00	26.0		II	4	5	L	Jobs loaded: 3
	2	III	5	18	3	1.00	26.0		III	3	5	L	Total off: 0
	3	III	5	17	0	1.04	21.2	L	IV	6	3	L,O	Total loaded: 3
	4	II	5	15	0	1.00	20.0	L	V	—	—		
	5	II	3	33	9	1.00	45.0		VI	—	—		
	5	III	9	33	0	1.63	25.8						
	6	II	3	27	0	1.20	25.0						
	6	IV	3	27	9	1.36		L					

Detailed sequencing (Continued)

TIME	JOBS ON OR IN QUEUE AT t_i								JOBS ON AT t_i				SUMMARY
t_i	P	J	M	S	F	D	f		J	P	M		
$t_1 = 3$	1	III	1	28	0	1.39			I	—	—		Jobs off this step: 1
$\Delta t = 3$	2	II	8	15	0	1.00			II	4	2	O	Jobs loaded: 0
	2	III	5	15	3	1.00			III	3	2	O	Total off: 1
	3	III	2	17	0	1.04		R	IV	—	—		Total loaded: 3
	4	II	2	15	0	1.00		R	V	—	—		
	5	II	3	30	9	1.00			VI	—	—		
	5	III	9	30	0	1.63							
	6	II	3	24	0	1.20							
$t_2 = 5$	1	III	1	26	0	1.39	19.4	L	I	4	5	L	Jobs off this step: 2
$\Delta t = 2$	2	II	8	13	0	1.00	21.0		II	6	3	L	Jobs loaded: 4
	2	III	5	13	3	1.00	21.0		III	1	1	L,O	Total off: 3
	3	IV	4	17	0	1.04		A,L	IV	3	4	L	Total loaded: 7
	4	I	5	15	0	1.00		A,L	V	—	—		
	5	II	3	28	9	1.00	40.0		VI	—	—		
	5	III	9	28	0	1.63	22.7						
	6	II	3	22	0	1.20	20.8	L					
$t_3 = 6$	1	I	3	26	0	1.47		A	I	4	4		Jobs off this step: 1
$\Delta t = 1$	2	II	8	12	0	1.00			II	6	2	O	Jobs loaded: 1
	2	III	5	12	0	1.00	17.0	L	III	2	5	L	Total off: 4
	3	IV	3	17	0			R	IV	3	3		Total loaded: 8
	4	I	4	15	0			R	V	—	—		
	5	II	3	27	9	1.00			VI	—	—		
	5	III	9	27	0	1.63	22.1						
	6	II	2	22	0	1.20		R					
$t_4 = 8$	1	I	3	24	0	1.47			I	4	2		Jobs off this step: 1
$\Delta t = 2$	2	II	8	10	0	1.00	18.0	L	II	2	8	L	Jobs loaded: 2
	2	III	3	10				R	III	2	3	R	Total off: 5
	3	IV	1	17				R	IV	3	1	O	Total loaded:10
	4	I	4	15				R	V	—	—		
	5	II	3	25	9	1.00	37.0		VI	6	9	L	
	5	III	9	25	0	1.63							
	6	VI	9	22	0	1.36		A,L					
$t_5 = 9$	1	I	3	23	0	1.47			I	4	1	O	Jobs off this step: 1
$\Delta t = 1$	2	II	7	10				R	II	2	7		Jobs loaded: 0
	2	III	2	10				R	III	2	2		Total off: 6
	3	VI	8	17	0	1.06		A	IV	—	—		Total loaded: 10
	4	I	4	15				R	V	—	—		
	5	II	3	24	9	1.00			VI	6	8		
	5	III	9	24	0	1.63							
	6	VI	8	22				R					
$t_6 = 10$	1	I	3	22	0	1.47		L	I	1	3	L	Jobs off this step: 1
$\Delta t = 1$	2	II	6	10				R	II	2	6		Jobs loaded: 1
	2	III	1	10				R	III	2	1	O	Total off: 7
	3	VI	8	16	0	1.06			IV	—	—		Total loaded: 11
	4	III	5	15	0	1.00		A	V	—	—		
	5	II	3	23	9	1.00			VI	6	7		
	5	III	9	23	0	1.63							
	6	VI	7	22				R					

Detailed sequencing (Continued)

TIME	JOBS ON OR IN QUEUE AT t_i								JOBS ON AT t_i				SUMMARY
t_i	*P*	*J*	*M*	*S*	*F*	*D*	*f*		*J*	*P*	*M*		
$t_7 = 11$	1	I	2	22	0	1.47		R	I	1	2	O	Jobs off this step: 1
	2	II	5	10				R	II	2	5		Jobs loaded: 1
$\Delta t = 1$	2	VI	10	10	3	1.00		A	III	4	5	L	Total off: 8
	3	VI	8	15	0	1.06			IV	—	—		Total loaded: 12
	4	III	5	14	0	1.00	19.0	L	V	—	—		
	5	II	3	22	9	1.00			VI	6	6		
	5	III	9	22	0	1.63	19.01						
	6	VI	6	22				R					
$t_8 = 13$	1	II	6	22	0	1.00		A	I	—	—		Jobs off this step: 1
$\Delta t = 2$	1	VI	7	22	2	1.00	31.0	A	II	2	3	O	Jobs loaded: 0
	2	II	3	10				R	III	4	3	O	Total off: 9
	2	VI	10	10	3	1.00	23.0		IV	—	—		Total loaded: 12
	3	VI	8	13	0	1.06	19.8		V	—	—		
	4	III	3	14	0	1.00		R	VI	6	4		
	5	II	3	20	9	1.00							
	5	III	9	20	0	1.63							
	6	VI	4	22				R					
$t_9 = 16$	1	II	6	19	0	1.00	25.0		I	—	—		Jobs off this step: 2
$\Delta t = 3$	1	VI	7	19	0	1.00	26.0		II	5	3	L	Jobs loaded: 4
	2	V	10	10	0	1.00		A,L	III	5	9	L	Total off: 11
	2	VI	10	10	0	1.00	20.0		IV	4	3	L	Total loaded: 16
	3	VI	8	10	0	1.06	16.9		V	2	10	L	
	4	IV	3	14	0	1.00		A,L	VI	6	1	O	
	5	II	3	17	0	1.00	20.0	L					
	5	III	9	17	0	1.63		L					
	6	VI	1	22				R					
$t_{10} = 17$	1	II	6	18	0	1.00			I	6	10	L	Jobs off this step: 1
$\Delta t = 1$	1	VI	7	18	0	1.00	25.0		II	5	2	O	Jobs loaded: 2
	2	V	9	9				R	III	5	8		Total off: 12
	2	VI	10	9	0	1.00	19.0		IV	4	2	O	Total loaded: 18
	3	VI	8	9	0	1.06	17.0	L	V	2	9		
	4	IV	2	14				R	VI	3	8	L	
	5	II	2	17				R					
	5	III	8	17				R					
	6	I	10	22	0	1.00		A,L					
	6	V	4	22	7	1.00		A					
$t_{11} = 19$	1	II	6	16	0	1.00		L	I	6	8		Jobs off this step: 2
$\Delta t = 2$	1	VI	7	16	0	1.00	23.0		II	1	6	L,O	Jobs loaded: 1
	2	V	7	7				R	III	5	6	O	Total off: 14
	2	VI	10	7	0	1.00	17.0		IV	—	—		Total loaded: 19
	3	VI	6	9				R	V	2	7		
	4	V	8	14	0	1.00		A	VI	3	6	O	
	5	III	6	17				R					
	6	I	8	22				R					
	6	V	4	22	5	1.00							

Detailed sequencing (Continued)

TIME	JOBS ON OR IN QUEUE AT t_i								JOBS ON AT t_i				SUMMARY
t_i	P	J	M	S	F	D	f		J	P	M		
$t_{12} = 25$	1	IV	3	12	4	1.00		A,L	I	6	2		Jobs off this step: 3
$\Delta t = 6$	1	VI	7	12	0	1.00	19.0		II	3	1	L,O	Jobs loaded 3
	2	V	1	1				R	III	—	—		Total off: 17
	2	VI	10	1	0	1.00	11.0	L	IV	1	3	L	Total loaded: 22
	3	I	9	9	0	1.00		A	V	2	1	O	
	3	II	1	9	15	1.00		A,L	VI	2	10	L	
	4	V	8	8	0	1.00	16.0						
	5	I	3	17	0	1.00		A					
	5	V	5	17	2	1.00	24.0	A					
	6	I	2	21				R					
	6	V	4	21	0	1.00	25.0						
$t_{13} = 26$	1	IV	2	11				R	I	6	1	O	Jobs off this step: 2
$\Delta t = 1$	1	VI	7	17	0	1.00			II	—	—		Jobs loaded: 1
	2	VI	9	1	0	1.00		R	III	—	—		Total off: 19
	3	I	9	8	0	1.00			IV	1	2		Total loaded: 23
	4	V	8	7	0	1.00	15.0	L	V	4	8	L	
	5	I	3	16	0	1.00			VI	2	9		
	5	V	5	16	2	1.00	23.0						
	6	I	1	20				R					
	6	V	4	20	0	1.00	24.0						
$t_{14} = 27$	1	IV	1	10				R	I	3	9	L	Jobs off this step: 1
$\Delta t = 1$	1	VI	7	10	0	1.00			II	—	—		Jobs loaded: 2
	2	VI	8	1	0	1.00		R	III	6	1	L,O	Total off: 20
	3	I	9	7	0	1.00	16.0	L	IV	1	1	O	Total loaded: 25
	4	V	7	7	0	1.00		R	V	4	7		
	5	I	3	15	0	1.00	18.0		VI	2	8		
	5	V	5	15	2	1.00							
	6	III	1	19	6	1.00		A,L					
	6	V	4	19	0	1.00							
$t_{15} = 28$	1	VI	7	9	0	1.00			I	3	8		Jobs off this step: 2
$\Delta t = 1$	2	VI	7	1				R	II	—	—		Jobs loaded: 0
	3	I	8	7				R	III	—	—		Total off: 22
	4	V	6	7				R	IV	—	—		Total loaded: 25
	5	I	3	14	0	1.00			V	4	6	O	
	5	V	5	14	2	1.00			VI	2	7		
	6	V	4	18	0	1.00							
$t_{16} = 34$	1	VI	7	3	0	1.00			I	3	2		Jobs off this step: 1
$\Delta t = 6$	2	VI	1	1				R	II	—	—		Jobs loaded: 1
	3	I	2					R	III	—	—		Total off: 23
	4	VI	9	7	0	1.00		A	IV	—	—		Total loaded: 26
	5	I	3	8	0	1.00			V	5	5	L	
	5	V	5	8	2	1.00	15.0	L	VI	2	1	O	
	6	V	4	12	0	1.00	16.0						

Detailed sequencing (Continued)

TIME	JOBS ON OR IN QUEUE AT t_i								JOBS ON AT t_i				SUMMARY
t_i	P	J	M	S	F	D	f		J	P	M		
$t_{17} = 35$	1	VI	7	2	0	1.00	0.0	L	I	3	1	O	Jobs off this step: 1
$\Delta t = 1$	2	I	10	1	0	1.00	11.0	A	II	—	—		Jobs loaded: 1
	3	I	1	7				R	III	—	—		Total off: 24
	4	VI	9	6	0	1.00	15.0		IV	—	—		Total loaded: 27
	5	I	3	7	0	1.00	10.0		V	5	4		
	5	V	4	7				R	VI	1	7	L	
	6	V	4	11	0	1.00							
$t_{18} = 36$	1	VI	6	2				R	I	5	3	L,O	Jobs off this step: 1
$\Delta t = 1$	2	I	10	0	0	1.00	10.0		II	—	—		Jobs loaded: 1
	3	V	7	7	0	1.00		A	III	—	—		Total off: 25
	4	VI	9	5	0	1.00			IV	—	—		Total loaded: 28
	5	I	3	6	0	1.00	9.0	L	V	5	3	O	
	5	V	3	6				R	VI	1	6		
	6	V	4	10	0	1.00							
$t_{19} = 39$	1	VI	3	2				R	I	2	10	L	Jobs off this step: 2
$\Delta t = 3$	2	I	10	−3	0	1.00		L	II	—	—		Jobs loaded: 2
	3	V	7	4	0	1.00	11.0	L	III	—	—		Total off: 27
	4	VI	9	2	0	1.00			IV	—	—		Total loaded: 30
	5	VI	4	6	0	1.00		A	V	3	7	L	
	6	V	4	7	0	1.00	11.0		VI	1	3	O	
$t_{20} = 42$	1	V	6	2	0	1.00		A	I	2	7		Jobs off this step: 1
$\Delta t = 3$	2	I	7	−3				R	II	—	—		Jobs loaded: 1
	3	V	4	4				R	III	—	—		Total off: 28
	4	VI	9	−1	0	1.00	9.0		IV	—	—		Total loaded: 31
	5	VI	4	3	0	1.00	7.0	L	V	3	4	O	
	6	V	4	4	0	1.00			VI	5	4	L,O	
$t_{21} = 46$	1	V	6	−2	0	1.00	6.0		I	2	3		Jobs off this step: 2
$\Delta t = 4$	2	I	3	−3	0	1.00		R	II	—	—		Jobs loaded: 3
	4	VI	9	−5				L	III	—	—		Total off: 30
	5	IV	1	3	0	1.00		A,L	IV	5	1	L,O	Total loaded: 34
	6	*V*	4	0	0	1.00	4.0	L	V	6	4	L	Project 3 finished
									VI	4	9	L	
$t_{22} = 47$	1	V	6	−3	0	1.00			I	2	2	O	Jobs off this step 1
$\Delta t = 1$	2	I	2	−3	0	1.00		R	II	—	—		Jobs loaded: 0
	4	VI	8	−5				R	III	—	—		Total off: 31
	6	V	3	0				R	IV	—	—		Total loaded: 34
									V	6	3		Project 5 finished
									VI	4	8		
$t_{23} = 49$	1	V	6	−5	0	1.00			I	—	—		Jobs off this step: 1
$\Delta t = 2$	2	IV	4	−3	0	1.00		A,L	II	—	—		Jobs loaded: 1
	4	VI	6	−5				R	III	—	—		Total off: 32
	6	V	1	0				R	IV	2	4	L	Total loaded: 35
									V	6	1	O	
									VI	4	6		

Detailed sequencing (Continued)

TIME	JOBS ON OR IN QUEUE AT t_i							JOBS ON AT t_i				SUMMARY
t_i	P	J	M	S	F	D	f	J	P	M		
$t_{24} = 50$	1	V	6	−6			L	I	—	—		Jobs off this step: 1
$\Delta t = 1$	2	IV	3	−3			R	II	—	—		Jobs loaded: 1
	4	VI	5	−5			R	III	—	—		Total off: 33
								IV	2	3	O	Total loaded 36
								V	1	6	L	Project 6 finished
								VI	4	5		
$t_{25} = 53$	1	V	3	−6			R	I	—	—		Jobs off this step: 1
$\Delta t = 3$	4	VI	2	−5			R	II	—	—		Total loaded: 0
								III	—	—		Total off: 34
								IV	—	—		Total loaded: 36
								V	1	3		Project 2 finished
								VI	4	2	O	
$t_{26} = 55$	1	V	1	−6			R	I	—	—		Jobs off this step: 1
$\Delta t = 2$								II	—	—		Jobs loaded: 0
								III	—	—		Total off: 35
								IV	—	—		Total loaded: 36
								V	1	1	O	Project 4 finished
								VI	—	—		
$t_{27} = 56$								I	—	—		Jobs off this step: 1
$\Delta t = 1$								II	—	—		Jobs loaded: 0
								III	—	—		Total off: 36
								IV	—	—		Total loaded: 36
								V	—	—		Project 1 finished
								VI	—	—		

Gantt Chart for six projects