MULTIPROJECT SCHEDULING WITH LIMITED RESOURCES: A ZERO-ONE PROGRAMMING APPROACH

A. Alan B. Pritsker
Lawrence J. Watters
Philip M. Wolfe

February 1968

P-3800

MULTIPROJECT SCHEDULING WITH LIMITED RESOURCES: A ZERO-ONE PROGRAMMING APPROACH

A. Alan B. Pritsker*

Consultant to The RAND Corporation, Santa Monica, California

Lawrence J. Watters*

The RAND Corporation, Santa Monica, California

Philip M. Wolfe

University of Arizona

*Any views expressed in this paper are those of the author. They should not be interpreted as reflecting the views of The RAND Corporation or the official opinion or policy of any of its governmental or private research sponsors. Papers are reproduced by The RAND Corporation as a courtesy to members of its staff.

MULTIPROJECT SCHEDULING WITH LIMITED RESOURCES: A ZERO-ONE PROGRAMMING APPROACH

A. Alan B. Pritsker, Arizona State University;
Consultant to The RAND Corporation, Santa Monica, California

Lawrence J. Watters, The RAND Corporation

Philip M. Wolfe, Arizona State University

ABSTRACT

A zero-one (0-1) linear programming formulation of multiproject and job-shop scheduling problems is presented that is more general and computationally tractable than other known formulations. It can accommodate a wide range of real-world situations including multiple resource constraints, due dates, job splitting, resource substitutability, and concurrency and nonconcurrency of job performance requirements. Three possible objective functions are discussed: minimizing total throughput time for all projects: minimizing the time by which all projects are completed (i.e., minimizing makespan); and minimizing total lateness or lateness penalty for all projects.

INTRODUCTION

Several years have passed since the pioneering work of Bowman [3], Wagner [24], and Manne [13] with their mathematical programming formulations of scheduling problems. These and other mathematical scheduling models are discussed by Sisson [22], by Conway, Maxwell, and Miller [4], and by Muth and Thompson [17]. Recent research efforts have concentrated on simulation approaches to scheduling [4,6,8,15,26,27]; however, another look at the problem from a mathematical programming point of view seems in order, especially in light of recent developments in 0-1 programming [7,9,18,25].

The scheduling problems considered here deal with determining when a job should be processed, given limited availabilities of resources, e.g., men, equipment, and facilities. The words job and project will be used throughout to denote the two levels of work aggregation being considered. A project consists of a set of jobs. In other literature describing scheduling research, the following equivalent descriptors may be found:

Job	Project
task	product
operation	job
project	program

The model considers only the job level and the project level. Consideration of three or more levels, e.g., operation, job, and project, only complicates the notational problem.

The Manne [13] formulation uses integer variables to indicate in which time period the job is started, and the Wagner [24] formulation uses 0-1 variables to indicate whether or not a job is assigned to a specific order-position on a specific machine. Neither formulation, however, accommodates multiple resource constraints. The Bowman [3] formulation uses 0-1 variables to indicate for each period over a scheduling horizon whether or not a job is being processed. His formulation does not expressly provide for multiple resource constraints, although such an extension could be made. The resulting formulation would be larger (in terms of the number of variables and constraints involved) than the one presented here. The following formulation uses

0-1 variables to indicate for select periods (depending upon job arrival time, due dates, sequencing relationships, etc.) whether or not a job is completed _in_ those periods. A similar formulation [19] uses 0-1 variables to indicate for select periods whether or not a job has been completed _prior to_ those periods. For a given type of scheduling environment, alternative formulations often may be devised. An efficient formulation, however, will depend upon a judicious choice of definition for the variables. Indeed, the selection of a definition for the variables and the synthesis of objective functions and constraints constitute a challenging problem in design. The design aspects of mathematical formulation are discussed in [20].

Determining when the jobs should be processed depends upon the desired objective. Three are considered here:

1. Minimize total throughput time (time in the shop) for all projects;
2. Minimize the time by which all projects are completed (i.e., minimize makespan); and
3. Minimize total lateness or lateness penalty for all projects.

Equations are developed to ensure that a schedule meets the following constraints when they are imposed:

1. Limited resources;
2. Precedence relations between jobs;
3. Job splitting possibilities;
4. Project and job due dates;
5. Substitution of resources to perform the jobs;
6. Concurrent and nonconcurrent job performance requirements,

DEFINITIONS

i = project number, $i = 1,2,\ldots,I$; I = number of projects.

j = job number, $j = 1,2,\ldots,N_i$; N_i = number of jobs in project i.

t = time period, $t = 1,2,\ldots \max G_i$; G_i = absolute due date. Project i must be completed in or before period G_i. If an absolute due date is not specified, G_i becomes the last period in the scheduling horizon.

g_i = desired due date. Project i is not late if it is completed in or before period g_i.

e_i = earliest possible period by which project i could be completed.

a_{ij} = arrival period of job j, project i. Arrivals occur at the beginning of periods.

d_{ij} = number of periods required to perform job j of project i. It is assumed to be known with certainty.

ℓ_{ij} = the earliest possible period in which job j could be completed.

u_{ij} = the latest possible period in which job j could be completed; viz., an absolute job due date.

k = resource or facility number, $k = 1,2,\ldots,K$; K = number of different resource types.

r_{ijk} = amount of type k resource required on job j of project i.

R_{kt} = amount of type k resource available in period t.

x_{ijt} = a variable which is 1 if job j of project i is completed in period t; 0 otherwise. x_{ijt} need not be treated as a variable in all periods, since it equals 0 for $t < \ell_{ij}$ and for $t > u_{ij}$.

x_{it} = a variable which is 1 in period t if all jobs of project i have been completed by period t (i.e., completed in or before period $t-1$), 0 otherwise. x_{it} need not be treated as a variable in all periods, since it equals 0 for $t < e_i$ and 1 for $t > G_i$.

To illustrate the nature of the definitions, the scheduling of five jobs belonging to two projects requiring two resources is shown in Fig. 1. The figure depicts arrival periods, job durations, due dates, precedence requirements, and values of x_{ijt} and x_{it} variables. There is one unit of resource available for each of the two types of resources; i.e., $R_{kt}=1$ for $k=1$, 2 and for all t. The resource requirements, r_{ijk}, for each job are assumed to be:

Resource Requirements, r_{ijk}

k \ ij	11	12	13	21	22
1	1	1	0	0	1
2	1	0	1	1	0

As an example of the calculations of ℓ_{ij} and u_{ij} for sequenced jobs,

$$u_{11} = G_1 - d_{13} = 11$$

and

$$\ell_{13}=\max\{a_{11}+d_{11}+d_{13}-1,\ a_{13}+d_{13}-1\} = 9.$$

The information depicted in Fig. 1 represents known inputs at the time of scheduling. (In a job-shop environment where additional projects arrive, rescheduling could take place when such inputs become known.)

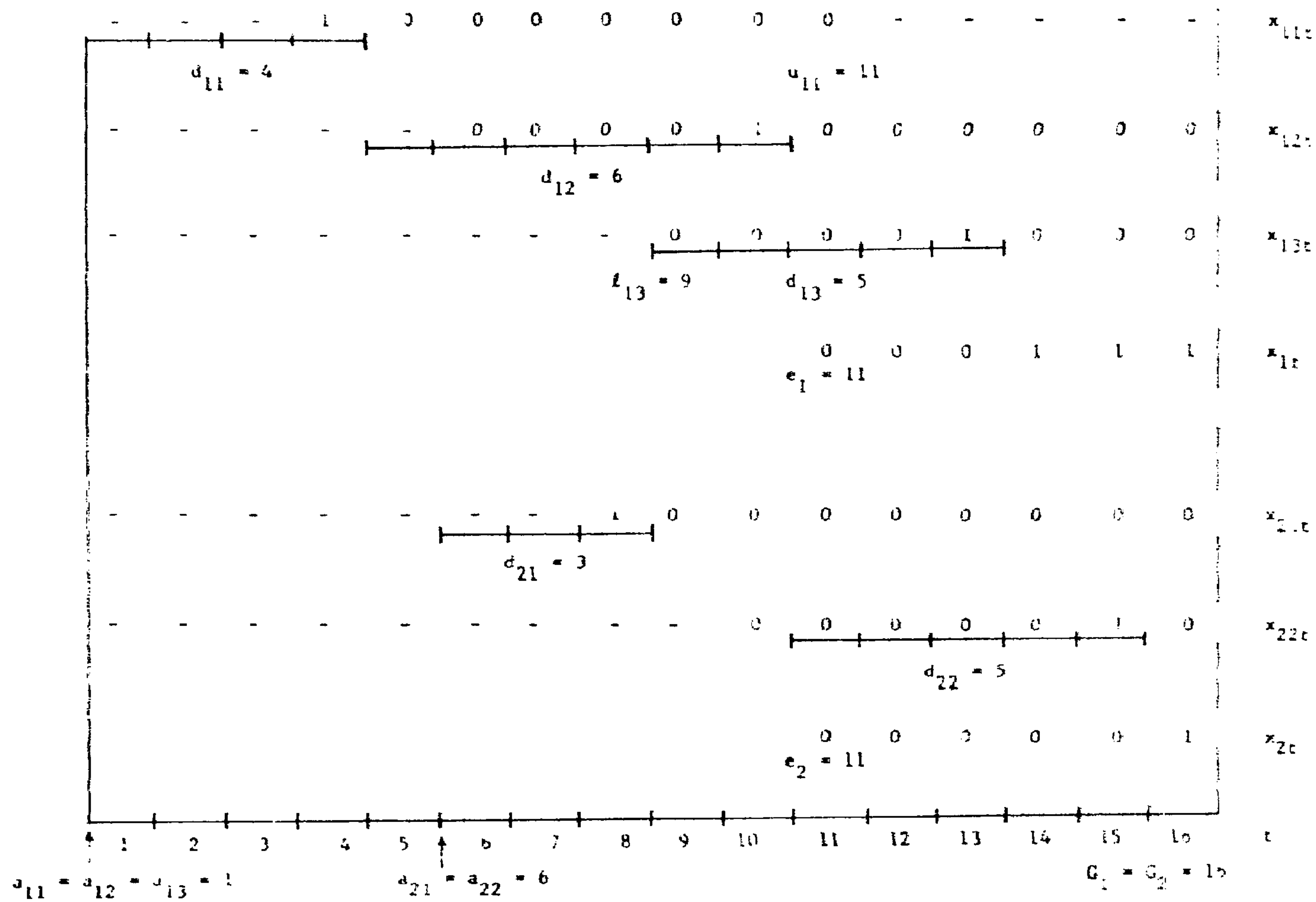

Fig. 1 -- Hypothetical Scheduling Situation for Two Projects

[Assume job (1,1) must precede job (1,3)]

(-) indicates the variable is predetermined to equal zero.

OBJECTIVE FUNCTIONS

Jobs are to be scheduled in a manner that optimizes some measure of performance, or objective function, subject to certain environmental requirements and limitations. The choice of an appropriate objective function may differ for various scheduling environments. Several common ones are selected for explicit formulation.

Minimizing Total Project Throughput Time

Individual project throughput time is defined as the elapsed time between project arrival and project completion, where project completion occurs when all jobs of the project are completed. If a_i is the arrival period of the i^{th} project, throughput time for that project is

$$G_i - \sum_{t=e_i}^{G_i} x_{it} + 1 - a_i .$$

(For example, throughput times for Projects 1 and 2 in Fig. 1 are 13 and 10, respectively.) Minimizing throughput time for a single project is equivalent to maximizing the number of periods remaining after the project is completed, where this number of periods is $\sum_{t=e_i}^{G_i} x_{it}$. Therefore, the objective function for minimizing the sum of the throughput times for all projects can be written as

$$\text{(1)} \qquad \text{Maximize } z = \sum_{i=1}^{I} \sum_{t=e_i}^{G_i} x_{it} .$$

Ordinarily, jobs are started as soon as possible if doing so does not increase throughput time. This can be accomplished by maximizing

(2)
$$z = \sum_{i=1}^{I} \sum_{t=e_i}^{G_i} x_{it} - \frac{1}{M} \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{t=\ell_{ij}}^{u_{ij}} t x_{ijt},$$

where M is a positive number sufficiently large to ensure that the contribution of the additional term is less than that of <u>any</u> x_{it}. A suitable choice for M would be

$$M > \sum_{i=1}^{I} \sum_{j=1}^{N_i} u_{ij}.$$

<u>Minimizing Makespan</u>

An alternative objective function is to minimize the time by which <u>all</u> projects are completed; i.e., minimize makespan. Define a variable x_t as follows:

$$x_t = \begin{cases} 1 & \text{if all projects are completed by period } t, \\ 0 & \text{otherwise.} \end{cases}$$

Minimizing makespan then corresponds to maximizing

(3)
$$z = \sum_{t=\max e_i}^{\max G_i} x_t .$$

This objective function could also be augmented to start jobs as soon as possible, thus making the desired objective function

$$z = \sum_{t=\max e_i}^{\max G_i} x_t - \frac{1}{M} \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{t=\ell_{ij}}^{u_{ij}} t x_{ijt}. \tag{4}$$

<u>Minimizing Total Lateness or Lateness Penalty</u>

A project is late if it is completed after the desired due-date period, g_i. Equivalently, the project is late if $x_{it} = 0$ in those periods t where $g_i < t \le G_i$. If total project lateness is to be minimized, this lateness can be written as

$$\sum_{i=1}^{I} \sum_{t=g_i+1}^{G_i} (1-x_{it}).$$

If a penalty of p_{it} is assessed when the project is not completed by period t, the total lateness penalty can be written as

$$\sum_{i=1}^{I} \sum_{t=g_i+1}^{G_i} p_{it}(1-x_{it}).$$

This expression for total lateness penalty reduces to total project lateness if all p_{it} are 1. Thus for both cases an equivalent form of the objective function is the maximization of

$$z = \sum_{i=1}^{I} \sum_{t=g_i+1}^{G_i} p_{it} x_{it}. \tag{5}$$

CONSTRAINTS

The formulation can accommodate a rather wide range of environmental requirements and limitations. Some of these are now discussed.

Job Completion

Each job has exactly one completion period.

$$(6) \qquad \sum_{t=\ell_{ij}}^{u_{ij}} x_{ijt} = 1 \qquad (i = 1,2,\ldots,I;\ j = 1,2,\ldots,N_i).$$

Notice that in each constraint, the value of any one x_{ijt} can be determined by the values of the others in that constraint. To use this relationship to full advantage, replace Constraint (6) by

$$(7) \qquad \sum_{t=\ell_{ij}}^{u_{ij}-1} x_{ijt} \le 1,$$

and define

$$x_{ij(u_{ij})} \equiv 1 - \sum_{t=\ell_{ij}}^{u_{ij}-1} x_{ijt}.$$

Replacing $x_{ij(u_{ij})}$ by its definitional equivalence can be used to reduce the total number of variables in the formulation.

Project Completion

Formulations involving x_{it} variables (e.g., objective functions (1), (2), and (5)) require that the x_{it} variables for each project be zero until all of its jobs have been completed. That is, project i cannot be completed by period t until $\sum_{q=\ell_{ij}}^{t-1} x_{ijq} = 1$ for all N_i jobs of project i. This requirement can be written as

$$(8) \qquad x_{it} \leq \frac{1}{N_i} \sum_{j=1}^{N_i} \sum_{q=\ell_{ij}}^{t-1} x_{ijq}$$

$$(i = 1,2,\ldots,I;\ t = e_i, e_i+1, \ldots, G_i).$$

In formulations involving x_t variables (e.g., objective functions (3) and (4)), the above constraints are replaced by

$$(9) \qquad x_t \leq \frac{1}{\sum_{i=1}^{I} N_i} \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{q=\ell_{ij}}^{t-1} x_{ijq}$$

$$(t = \max e_i, \ldots, \max G_i).$$

<u>Sequencing</u>

A sequencing constraint is required when a job cannot be started until one or more other jobs have been completed. For example, on project i, assume job m must precede job n. If t_{im} and t_{in} denote the completion periods of jobs m and n respectively, then

$$t_{im} + d_{in} \leq t_{in}.$$

Note that $t_{im} = \sum_{t=\ell_{im}}^{u_{im}} tx_{imt}$ and $t_{in} = \sum_{t=\ell_{in}}^{u_{in}} tx_{int}$ Therefore, the appropriate sequencing constraint becomes

(10) $$\sum_{t=\ell_{im}}^{u_{im}} tx_{imt} + d_{in} \le \sum_{t=\ell_{in}}^{u_{in}} tx_{int}.$$

Sequencing relationships reduce the number of x_{ijt} variables for which it is necessary to obtain values from the formulation, since

(1) $x_{int} = 0$ for $t < \max\{a_{in} + d_{in} - 1;\ \max\limits_{j \in P_{in}} (a_{ij} + d_{ij} + d_{in} - 1)\}$ where P_{in} is the set containing other jobs of project i that must precede job n, and

(2) $x_{imt} = 0$ for $t > \min\limits_{j \in F_{im}} \{G_i - d_{ij}\}$ where F_{im} is the set containing other jobs of project i that must follow job m.

The number of x_{it} variables might also be reduced, since e_i might be increased as a result of sequencing relationships.

Resource Constraints

The value r_{ijk} specifies the number of resource units of type k required for the performance of job j, project i. Thus if $r_{ij1} = 3$ and $r_{ij2} = 2$, then 3 units of type 1 are used in conjunction with 2 units of type 2 during those periods when the job is being performed. Resources required on a job are assumed in use until the job ends. If this assumption is not appropriate, slight reformulation is required. For example, if a certain resource is in use only during the first p periods of the job where $p < d_{ij}$, then treat the job as two sequenced "subjobs" with differing resource requirements and with durations of p and $d_{ij} - p$, respectively. If the subjobs are to be performed contiguously, replace the $\le$ by = in Constraint (10). The approach can apply to any division of a job into two or more subjobs.

In any given period, the amount of resource k used on all jobs cannot exceed the amount of resource k available. A job is being processed in period t if the job is completed in period q where $t \le q \le t + d_{ij} - 1$. Therefore the resource constraint can be written as

$$\sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{q=t}^{t+d_{ij}-1} r_{ijk} x_{ijq} \le R_{kt} \qquad (11)$$

$$(t = \min a_{ij}, \ldots, \max G_i;\ k = 1,2,\ldots,K).$$

Implementation of this constraint necessitates recognizing predetermined values of x_{ijt}. (Namely, $x_{ijt} \equiv 0$ for $t < \ell_{ij}$ and $x_{ijt} \equiv 0$ for $t > u_{ij}$).

If the availability of a resource is constant over the scheduling horizon, then some periods may involve redundant resource constraints. Using the scheduling situation described by Fig. 1 as an example, the resource constraints associated with periods 1, 2, and 3 (viz., $t < \min\{a_{ij} + d_{ij}\} - 1$) would be redundant with those of period 4, and therefore removable. A more general observation is that if the resource availability, R_{kt}, is constant for the first t' periods where $\min_{i,j \ni r_{ijk}>0} \{a_{ij} + d_{ij}\} \le t' \le \max G_i$, then Constraint (11) need not be imposed for periods $t < \min_{i,j \ni r_{ijk}>0} \{a_{ij} + d_{ij}\} - 1$. On the other hand if R_{kt} is constant for the first t' periods where $\min_{i,j \ni r_{ijk}>0} \{a_{ij}\} \le t' \le \min_{i,j \ni r_{ijk}>0} \{a_{ij} + d_{ij}\}$, then Constraint (11) need not be imposed for periods $t < t' - 1$.

Substitutability of Resources

It may be possible to use alternative resources to accomplish some jobs. For example a man with a higher skill can be substituted for a man with a lower skill on particular jobs.

If resource substitution is permitted on job j, project i, then Constraint (11) must be modified to account for the resource substitution and potential differences in job durations when the job is performed by different resources. To handle this condition, define the job duration when done by resource k as d_{ijk}. Only the case where the job can be done with either resource k_1 with duration d_{ijk_1}, or k_2 with duration d_{ijk_2} will be considered. Both k_1 and k_2 could be considered as resource sets in which case each constraint involving the resources of the resource set requires modification. Assume $d_{ijk_1} < d_{ijk_2}$. Let

$$\delta_{ij} = \begin{cases} 0 & \text{if resource } k_2 \text{ is used} \\ 1 & \text{if resource } k_1 \text{ is used} \end{cases}$$

and add the following constraint:

$$(12) \qquad (1 - \delta_{ij}) \sum_{t=a_{ij}+d_{ijk_1}-1}^{a_{ij}+d_{ijk_2}-2} x_{ijt} = 0.$$

Constraint (12) requires $x_{ijt} = 0$ for $a_{ij} + d_{ijk_1} - 1 \leq t \leq a_{ij} + d_{ijk_2} - 2$ if $\delta_{ij} = 0$ (i.e., it prevents the job completion from occurring prior to period $a_{ij} + d_{ijk_2} - 1$ if resource k_2 is used), but does not restrict x_{ijt} if $\delta_{ij} = 1$. Constraint (11) must be altered to indicate whether resource k_1 or k_2 is employed to perform the job. For resources k_1 and k_2, the term in Constraint (11) associated with the job becomes

$$(13) \qquad \delta_{ij} \sum_{q=t}^{t+d_{ijk_1}-1} r_{ijk_1} x_{ijq}$$

for resource k_1, and

$$(14) \qquad (1 - \delta_{ij}) \sum_{q=t}^{t+d_{ijk_2}-1} r_{ijk_2} x_{ijq}$$

for resource k_2.

The above constraints can be put into a linear form by defining

$$y_{ijt} = \begin{cases} 1 & \text{if } \delta_{ij} x_{ijt} = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Using a technique Watters [25] developed, the following constraints are obtained to insure that the conditions imposed on y_{ijt} are satisfied:

$$(15) \qquad y_{ijt} \geq \delta_{ij} + x_{ijt} - 1, \quad \text{and}$$

$$(16) \qquad y_{ijt} \leq \tfrac{1}{2}(\delta_{ij} + x_{ijt}).$$

Constraint (12), in terms of y_{ijt}, becomes

$$(17)\qquad \sum_{t=a_{ij}+d_{ijk_1}-1}^{a_{ij}+d_{ijk_2}-2} (x_{ijt} - y_{ijt}) = 0.$$

Relationships (13) and (14) become

$$(18)\qquad \sum_{q=t}^{t+d_{ijk_1}-1} r_{ijk_1} y_{ijq} \qquad \text{and}$$

$$(19)\qquad \sum_{q=t}^{t+d_{ijk_2}-1} r_{ijk_2} (x_{ijq} - y_{ijq}).$$

Another approach to permitting a resource substitution is to define a set of mutually exclusive jobs, only one of which must be performed. For example, if two alternative methods (or resource combinations) exist for performing job j', define the two alternatives as jobs j_1 and j_2 with durations d_{ij_1} and d_{ij_2}, and with $u_{ij_1} = u_{ij_2} = u_{ij}'$. Require completion of either j_1 or j_2, but not both, anytime before the end of period u_{ij}'. To do this, replace Constraint (7) by

$$(20)\qquad \sum_{q=\min\{\ell_{ij_1}, \ell_{ij_2}\}}^{u_{ij}'} (x_{ij_1q} + x_{ij_2q}) = 1,$$

and retain $x_{ij(u_{ij}')}$ in the formulation (i.e., do not replace $x_{ij(u_{ij}')}$ by its definitional equivalent). The modified project completion constraint corresponding to Constraint (8) would be

$$(21) \qquad x_{it} < \frac{1}{N_i}\left[\sum_{\substack{j=1\\ j\neq j'}}^{N_i}\ \sum_{q=\ell_{ij}}^{t-1} x_{ijq} + \sum_{q=\min(\ell_{ij_1},\ell_{ij_2})}^{t-1} (x_{ij_1q} + x_{ij_2q})\right].$$

A similar modification could be made to Constraint (9).

Concurrency and Nonconcurrency of Jobs

A concurrency constraint on jobs m and n ensures that they must be performed simultaneously. It can be obtained by requiring $x_{imt} = x_{int}$, or by combining resource requirements and treating m and n as a single job.

A nonconcurrency constraint on jobs m and n ensures that they must not be performed simultaneously, but permits them to be performed in any order. Job m is being performed in period t if and only if

$$\sum_{q=t}^{t+d_{im}-1} x_{imq} = 1,$$

and similarly for job n. Thus the desired constraint is

$$(22) \qquad \sum_{q=t}^{t+d_{im}-1} x_{imq} + \sum_{q=t}^{t+d_{in}-1} x_{inq} \leq 1$$

$$(t = \max(\ell_{im},\ell_{in}),\ldots,\min(u_{im},u_{in})).$$

Job Splitting

Theoretically, total job-splitting capability could be accomplished by treating each job as d_{ij} subjobs (each subjob having one period duration) and by imposing appropriate sequencing constraints on these subjobs. Pragmatically, however, job-splitting capability would seldom be fully exercised because of setup costs, the desirability of maintaining job continuity, etc. Hence, defining substantially fewer than d_{ij} subjobs for a particular job may provide sufficient splitting flexibility without requiring an inordinate number of subjobs.

Suppose job j can be split, and its subjobs are sequenced in accordance with Constraint (10). When two of its sequenced subjobs, say m and n, are not performed contiguously (i.e., when the larger job of which they are a part is allowed to split), then

$$(23) \qquad \tau_{mn} = \sum_{t=\ell_{in}}^{u_{in}} tx_{int} - \sum_{t=\ell_{im}}^{u_{im}} tx_{imt} - d_{in}$$

represents the duration of the split. τ_{mn} is the slack variable of Constraint (10).

Penalty Costs. If a penalty cost, c_n, is incurred per time period of split, then $c_n\tau_{mn}$ is the job-splitting penalty cost. If a penalty cost, C_n, is incurred for the split regardless of split duration, then $C_n\tau_n$ is the job-splitting penalty cost where τ_n is a 0-1 variable such that

$$\tau_n = \begin{cases} 1 & \text{if a split occurs; i.e., } \tau_{mn} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

The appropriate value of τ_n is obtained by requiring

$$(24) \qquad \tau_n \geq \tau_{mn}/G_i \qquad \text{and}$$

$$\tau_n \le 1 + \frac{\tau_{mn} - 1}{G_i} . \qquad (25)$$

Both types of job-splitting penalty costs can be used with a cost objective function.

Duration Extension. If net job duration increases as the result of a split, the τ_n variable can be used to modify the resource constraints using a technique similar to that used in considering resource substitutions. If w_{in} is the duration penalty when subjob n does not immediately follow subjob m, then the terms in the appropriate resource constraints for subjob n become

$$\left[(1 - \tau_n) \sum_{q=t}^{t+d_{in}-1} x_{inq} + \tau_n \sum_{q=t}^{t+d_{in}+w_{in}-1} x_{inq}\right] r_{ink} . \qquad (26)$$

Again the quadratic terms would have to be replaced by their linear equivalents as described previously.

EXAMPLE SCHEDULING PROBLEM

A three-project, eight-job, three-resource-type problem will be formulated using the job completion, project completion, sequencing, and resource constraints. Jobs are to be scheduled so as to minimize total project throughput time. In addition, jobs are to be started as soon as possible if doing so does not increase total project throughput time. Table 1 contains sequencing relationships, job arrival and duration times, due dates, resource requirements, and resource availabilities. For comparative purposes, solutions provided by several standard dispatching rules are presented.

Table 1

SEQUENCING, ARRIVAL TIMES, JOB DURATIONS, DUE DATES, AND RESOURCE REQUIREMENTS

Project (i)	Job (j)	Precedence Relations (i,j)	Arrival Time (a_{ij})	Duration (d_{ij})	Absolute Due Date (G_i)	Resource Requirements (r_{ijk})		
						$k = 1$	$k = 2$	$k = 3$
1	1	None	1	4	8	5	3	2
1	2	(1,1)	1	3	8	0	1	1
1	3	None	1	3	8	2	0	2
2	1	None	2	3	9	1	1	1
2	2	None	2	2	9	2	0	0
2	3	(2,1)	2	2	9	2	2	0
3	1	None	3	5	9	2	1	1
3	2	None	3	1	9	1	3	0
Amount of resource k available in each time period (R_{kt})						8	5	4

Dispatching Rules

Resource requirements for jobs and limited resource availability preclude jobs from being started immediately. Immediate dispatch, as depicted by the schedule in Fig. 2, would cause resource requirements to exceed resource availability. As seen from Fig. 2, minimum throughput times for Projects 1, 2, and 3 are seven time units, five time units, and five time units, respectively. Thus, any feasible solution to the scheduling problem could not yield a total throughput time of less than 17 time units. The following dispatch examples observe arrival and sequencing constraints as resource conflicts are resolved. In some cases, the due dates are not met.

First-Come-First-Served. Figure 3 indicates a schedule obtained when the jobs are processed on a first-come-first-served basis, arrival

	Resource Usage									Resource Availability (R_{kt})
k = 1	7	10	13	8	4	4	2	0	0	8
k = 2	3	4	8	5	4	4	2	0	0	5
k = 3	4	5	6	4	2	2	2	0	0	4

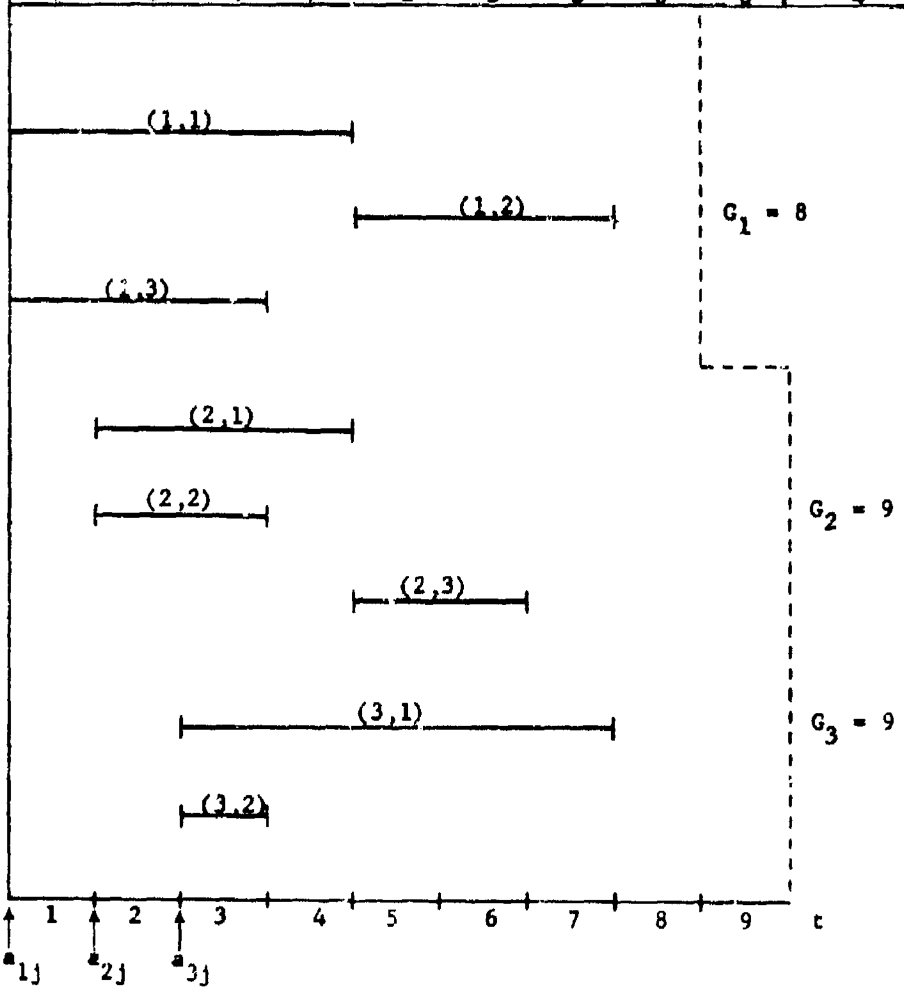

Fig. 2 -- Earliest start and completion times, unlimited resources

ties are broken by processing the shortest job first. This rule produces a schedule for completing Projects 1 and 2 on time, but Project 3 is late by one time unit. Total throughput time is 22 time units. If ties are broken by processing the longest job first instead of the shortest job, the schedule in Fig. 4 results. No lateness occurs, and total throughput time is 21 time units.

Minimum-Project-Slack-First. Priority is determined by project slack (the time between the earliest and the latest permissible project completion time). From Fig. 2,

Project 1 slack = one time unit;
Project 2 slack = three time units;
Project 3 slack = two time units.

Therefore, under the minimum-project-slack-first dispatch rule, jobs of Project 1 are scheduled first, then those of Project 3, and finally those of Project 2. Figure 5 depicts the resulting schedule. Projects 1 and 3 are completed on time, but Project 2 is late by three time units. Total throughput time is 24 time units.

Formulation of Example

Variables required in the formulation are numbered as follows:

Variable No.	1	2	3	4	5	6	7	8	9	10
Variable	x_{114}	x_{127}	x_{133}	x_{134}	x_{135}	x_{136}	x_{137}	x_{214}	x_{215}	x_{216}
Variable No.	11	12	13	14	15	16	17	18	19	20
Variable	x_{223}	x_{224}	x_{225}	x_{226}	x_{227}	x_{228}	x_{236}	x_{237}	x_{238}	x_{317}
Variable No.	21	22	23	24	25	26	27	28	29	30
Variable	x_{318}	x_{323}	x_{324}	x_{325}	x_{326}	x_{327}	x_{328}	x_{18}	x_{27}	x_{28}
Variable No.	31	32	33							
Variable	x_{29}	x_{38}	x_{39}							

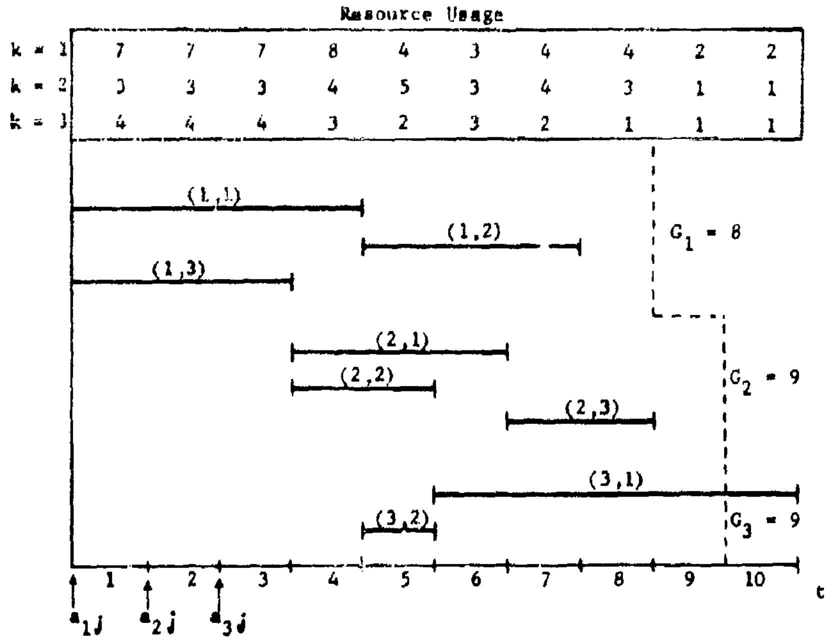

Fig. 3 -- FCFS, breaking ties with shortest job first

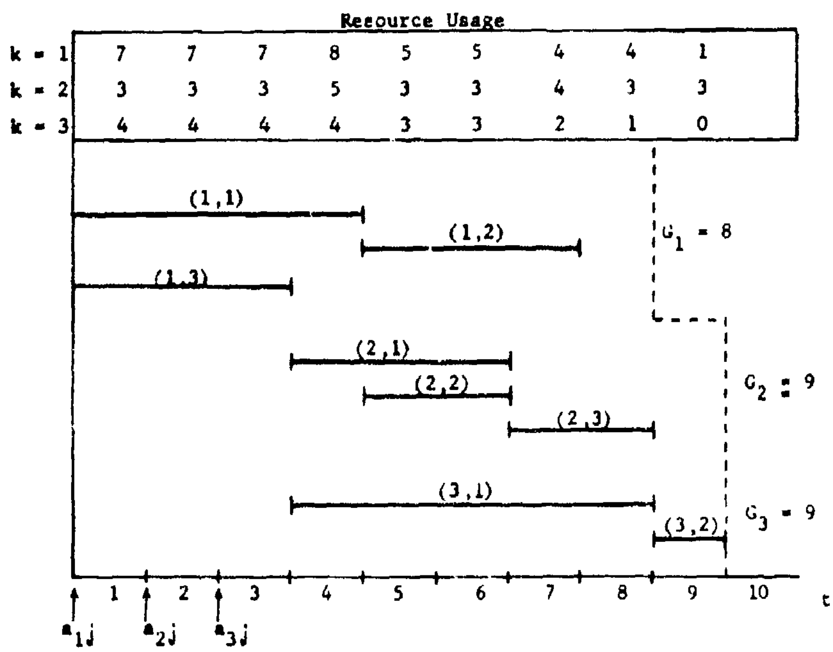

Fig. 4 -- FCFS, breaking ties with longest job first

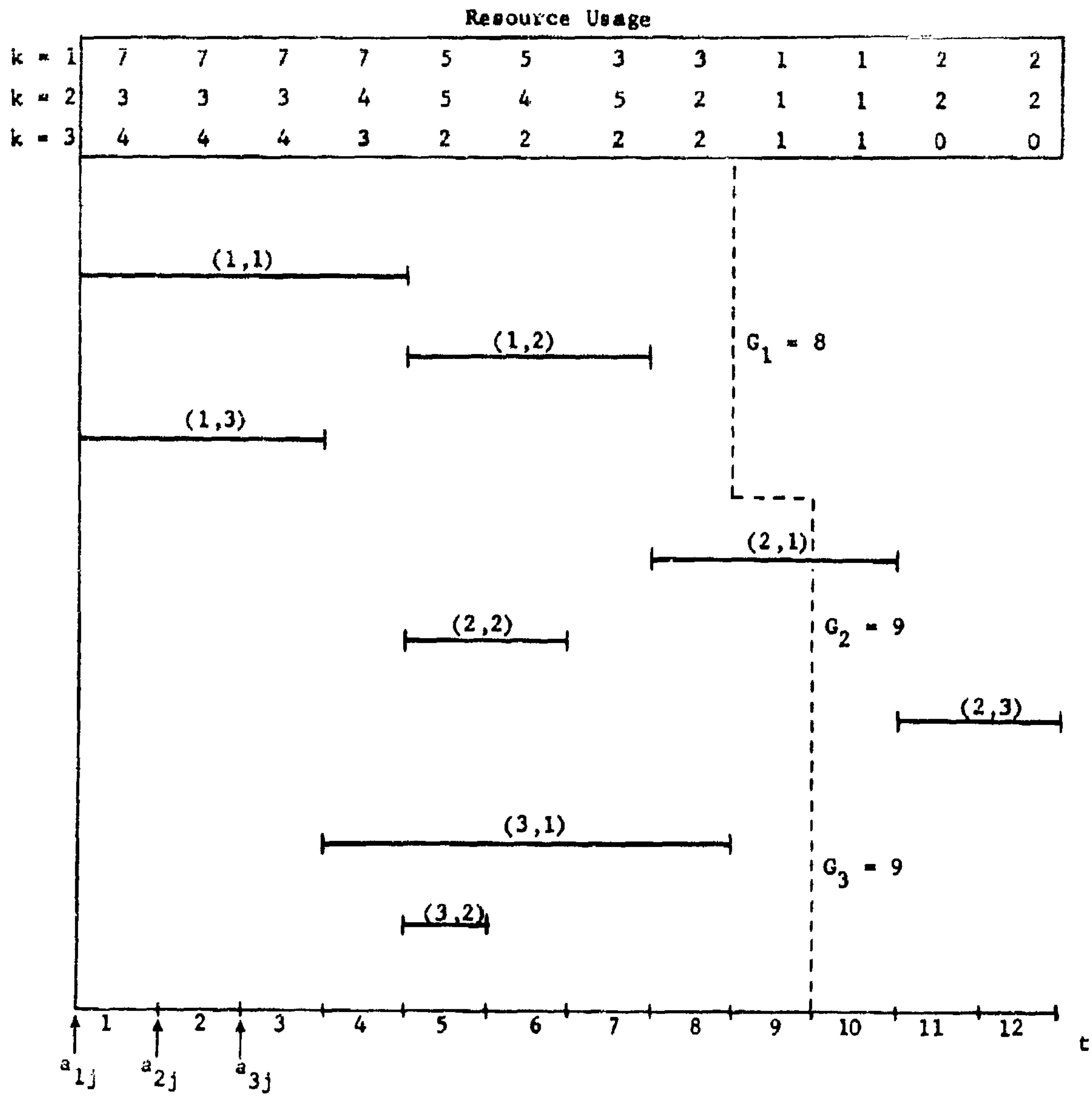

Fig. 5 -- Minimum project slack first

All $x_{ij(u_{ij})}$ variables (viz., x_{115}, x_{128}, x_{138}, x_{217}, x_{229}, x_{239}, x_{319}, x_{329}) are expressed in terms of their definitional equivalents.

Maximizing objective function (2) provides minimum total throughput time and starts jobs as soon as possible without otherwise affecting throughput time, provided

$$M > \sum_{i=1}^{I} \sum_{j=1}^{N_i} u_{ij} = 64.$$

The value $M = 65$ will be used.

Table 2 contains the coefficients of the objective function (multiplied by M so that they are all integer) and the constraints. The constraints are arranged as follows:

1-8 are job completion constraints;
9-14 are project completion constraints;
15-16 are sequencing constraints; and
17-37 are resource constraints for periods
$t \geq \min\{a_{ij} + d_{ij}\} - 1 = 3.$

Note that some constraints in Table 2 are nonbinding (viz., 1, 2, 34, 35, and 37) and may be deleted from the formulation.

<u>Solution Using a 0-1 Code</u>

The problem was solved with a 0-1 integer linear programming code developed by Geoffrion [7] and programmed for RAND's IBM 7044. Execution time is 2.3 seconds. The optimal solution is presented in Table 3.

Table 2

OBJECTIVE FUNCTION AND CONSTRAINT COEFFICIENTS

Constraint Number	Variable Number 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Constraint
Objective Function Coefficient	-4	-7	-3	-4	-5	-6	-7	-4	-5	-6	-3	-4	-5	-6	-7	-8	-6	-7	-8	-7	-8	-3	-4	-5	-6	-7	-8	65	65	65	65	65	65	
1	1																																	≤ 1
2		1																																≤ 1
3			1	1	1	1	1																											≤ 1
4								1	1	1																								≤ 1
5											1	1	1	1	1	1																		≤ 1
6																	1	1	1															≤ 1
7																				1	1													≤ 1
8																						1	1	1	1	1	1							≤ 1
9		-1	-1	-1	-1	-1	-1																					3						≤ 1
10								-1	-1	-1	-1	-1	-1	-1			-1												3					≤ 0
11											-1	-1	-1	-1	-1		-1	-1												3				≤ 1
12											-1	-1	-1	-1	-1	-1	-1	-1	-1												3			≤ 1
13																				-1		-1	-1	-1	-1	-1						2		≤ 0
14																				-1	-1	-1	-1	-1	-1	-1	-1						2	≤ 0
15	-1	1																																≤ 0
16								-3	-2	-1							3	2	1															≤ 0
17			2	2	2			1	1		2	2								2		1												≤ 3
18								1	1											1		3												≤ 2
19			2	2	2			1	1											1														≤ 2
20				2	2	2		1	1	1		2	2							2	2		1											≤ 3
21								1	1	1										1	1		3											≤ 2
22				2	2	2		1	1	1										1	1													≤ 2
23	-5				2	2	2	-1					2	2			2							1										≤ 0
24	-3	1						-1									2							3										≤ 0
25	-2	1			2	2	2	-1																										≤ 0
26			-2	-2	-2			-1	-1					2	2		2	2							1									≤ 3
27								-1	-1								2	2							3									≤ 2
28			-2	-2	-2			-1	-1																									≤ -1
29			-2	-2	-2	-2		-1	-1	-1					2	2		2	2							1								≤ 3
30								-1	-1	-1								2	2							3								≤ 2
31			-2	-2	-2	-2		-1	-1	-1																								≤ -1
32			-2	-2	-2	-2	-2				-2	-2	-2	-2	-2		-2	-2		-2							1							≤ 0
33		-1															-2	-2		-1							3							≤ 1
34		-1	-2	-2	-2	-2	-2													-1														≤ 0
35											-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-1	-1	-1	-1	-1	-1							≤ 1
36																	-2	-2	-2	-1	-1	-3	-3	-3	-3	-3	-3							≤ -1
37																				-1	-1													≤ 3

Table 3

SOLUTION TO EXAMPLE

Variable \ t	1	2	3	4	5	6	7	8	9
x_{11t}	-	-	-	1	0	-	-	-	
x_{12t}	-	-	-	-	-	-	1	0	
x_{13t}	-	-	-	0	0	0	1	0	
x_{1t}	-	-	-	-	-	-	-	1	
x_{21t}	-	-	-	1	0	0	0	-	-
x_{22t}	-	-	-	0	0	1	0	0	0
x_{23t}	-	-	-	-	-	1	0	0	0
x_{2t}	-	-	-	-	-	-	1	1	1
x_{31t}	-	-	-	-	-	-	1	0	0
x_{32t}	-	-	-	0	0	0	1	0	0
x_{3t}	-	-	-	-	-	-	-	1	1

(-) indicates the variable is predetermined to equal zero.

The optimum schedule thus determined is presented in Fig. 6. Projects 1, 2 and 3 are completed one, three, and two time periods ahead of their respective due dates. Total throughput time is 17 time units.

For this example the mathematical programming solution represents a substantial improvement over the solutions obtained from the first-come-first-served and minimum-project-slack-first dispatch rules. One dispatch rule that did yield the optimal solution was a minimum-job-

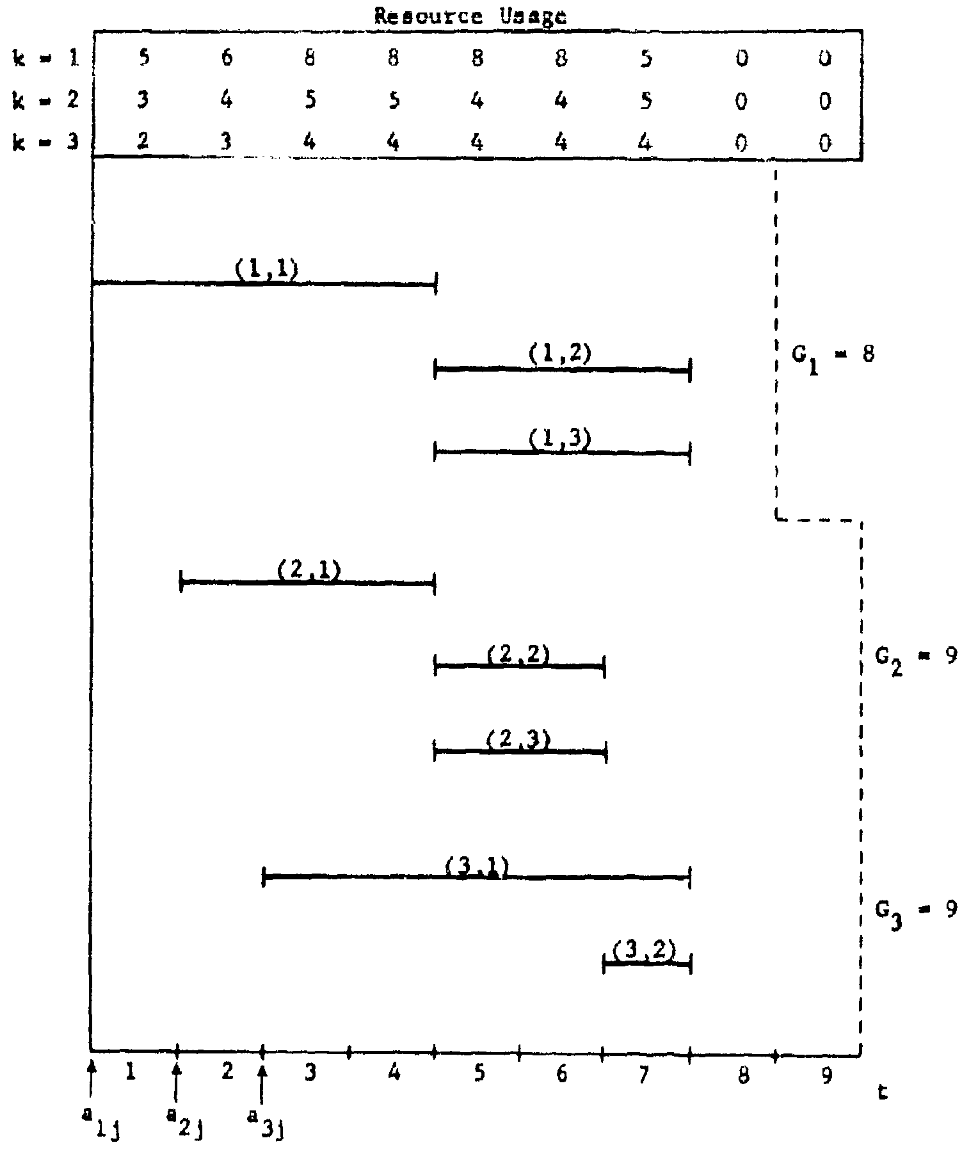

Fig. 6 -- Optimal solution

slack-first rule that determines priority as the time between the earliest and the latest permissible job completion time. However, no attempt was made to test or evaluate dispatch rules exhaustively.

This problem, when formulated in terms of the variables Bowman [3] uses and extended to accommodate multiple resources, would involve 72 variables and 125 constraints. If predetermined variables are eliminated, the Bowman formulation could be reduced to 50 variables and 94 constraints, still larger than the 33-variable, 37-constraint formulation presented here.

CONCLUSION

A zero-one linear programming formulation of scheduling problems has been developed which can accommodate a wide range of conditions. The formulation is more efficient than previously reported models in terms of the number of variables and the number of constraints required to model a scheduling situation. One general comment on the size of the formulation is that it is favorably affected by an increased amount of sequencing, by relatively long jobs, and by close proximity of the scheduling horizon (or absolute due date) to the optimal project completion date. This research coupled with the immense research on zero-one programming codes should yield practical procedures for obtaining optimal solutions to scheduling problems.

REFERENCES

1. Balas, E., "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," Operations Research, Vol. 13, 1965, pp. 517-546.

2. Balinski, M. L., "Integer Programming: Methods, Uses, Computation," Management Science, Vol. 12, No. 3, November 1965, pp. 253-313.

3. Bowman, E. H., "The Schedule-Sequencing Problem," Operations Research, Vol. 7, 1959, pp. 621-624.

4. Conway, R. W., W. L. Maxwell, and L. W. Miller, Theory of Scheduling, Addison-Wesley Publishing Co., Inc., Reading, Massachusetts, 1967.

5. Davis, E. W., "Resource Allocation in Project Network Models--A Survey," Journal of Industrial Engineering, Vol. 17, April 1966, pp. 177-188.

6. Fendley, L. G., "The Development of a Complete Multi-Project Scheduling System Using a Forecasting and Sequencing Technique," Ph.D. Dissertation, Arizona 1959. University, 1967.

7. Geoffrion, A. M., Implicit Enumeration Using an Imbedded Linear Program, The RAND Corporation, RM-5406-PR, September 1967.

8. Gere, W., "Heuristics in Job-Shop Scheduling," Management Science, Vol. 13, No. 3, November 1966, pp. 167-190.

9. Glover, F., "A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem," Operations Research, Vol. 13, 1965, pp. 879-919.

10. Hillier, F. S., Chance-Constrained Programming with Zero-One or Bounded Decision Variables, Technical Report 4, Contract NONR-225 (89), Stanford University, Stanford, California, August, 1966.

11. Lambourn, S., "RAMPS--A New Tool in Planning and Control," The Computer Journal, Vol. 5, 1963, pp. 300-304.

12. Lawler, E. L., "On Scheduling Problems with Deferral Costs," Management Science, Vol. 11, No. 2, November 1964, pp. 280-288.

13. Manne, A. S., "On the Job-Shop Scheduling Problem," Operations Research, Vol. 8, 1960, pp. 219-223.

14. McNaughton, Robert, "Scheduling with Deadlines and Loss Functions," Management Science, Vol. 6, No. 1, October State

15. Mize, J. H., "A Heuristic Scheduling Model for Multi-Project Organizations," Ph.D. Thesis, Purdue University, 1964.

16. Moder, J. J., and C. R. Phillips, Project Management With CPM and PERT, Reinhold Publishing Company, New York, 1964.

17. Muth, J. F., and G. L. Thompson (eds.), Industrial Scheduling, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1963.

18. Petersen, C. C., "Computational Experience with Variants of the Balas Algorithm Applied to the Selection of R&D Projects," Management Science, Vol. 13, 1967, pp. 736-745.

19. Pritsker, A.A.B., and L. J. Watters, A Zero-One Programming Approach to Scheduling with Limited Resources, The RAND Corporation, RM-5561-PR, January 1968.

20. Pritsker, A.A.B., L. J. Watters, and P. M. Wolfe, "Mathematical Formulation: A Problem in Design," Proceedings of the 19th Annual Institute Conference of AIIE, Tampa, Florida, May 1968.

21. Root, J. G., "Scheduling with Deadlines and Loss Functions on k Parallel Machines," Management Science, Vol. 11, No. 3, January 1965, pp. 460-475.

22. Sisson, R. L., "Sequencing Theory" in Progress in Operations Research, Vol. I, R. L. Ackoff (ed.), John Wiley and Sons, Inc., New York, 1961.

23. Smith, R. D., and R. A. Dudek, "A General Algorithm for Solution of the n-Job, M-Machine Sequencing Problem of the Flow Shop," Operations Research, Vol. 15, No. 1, January-February 1967, pp. 71-82.

24. Wagner, H., "An Integer Linear Programming Model for Machine Scheduling," Naval Research Logistics Quarterly, Vol. 6, 1959, pp. 131-140.

25. Watters. L. J., "Reduction of Integer Polynomial Programming Problems to Zero-One Linear Programming Problems," Operations Research, Vol. 15, No. 6, November-December 1967, pp. 1171-1174.

26. Wiest, J. D., "Some Properties of Schedules for Large Projects with Limited Resources," Operations Research, Vol. 12, May-June 1964, pp. 395-418.

27. -----, "A Heuristic Model for Scheduling Large Projects with Limited Resources," Management Science, Vol. 13, No. 6, February 1967, pp. B-359-377.