Resource-Constrained Scheduling of Projects with Variable-Intensity Activities

ROBERT C. LEACHMAN
Senior Member, IIE
Engineering Systems Research Center
University of California at Berkeley
Berkeley, CA 94720

ABDURREZAK DINCERLER
Muhas Inc.
Istanbul, Turkey

SOOYOUNG KIM
Member, IIE
Department of Industrial Engineering
Rutgers University
Piscataway, NJ 08855

Abstract: We consider resource-constrained scheduling of projects including activities whose resource loading is flexible. We propose a model of project-oriented production in which the application rates of the various resources required by an activity are indexed by the performance speed or *intensity* of the activity. Heuristic algorithms are introduced for the assignment of activity intensities through time in lieu of traditional, fixed-intensity start-time scheduling. On test problems from the literature, the assignment algorithms are shown to outperform even optimal scheduling algorithms for the fixed-intensity case, yet the assignment algorithms are computationally practical for scheduling actual, large-scale industrial projects. We also extend the model and the assignment algorithms to admit production-like workflow dependencies and to interface with aggregate models for multi-project planning.

■ Since CPM and PERT were introduced in late 50's, there have been numerous treatments of multiple resource-constrained project scheduling problems. Most models proposed for this type of problem assume that each activity in the project must be performed in one prespecified way, i.e., resources are applied to activities at fixed rates and durations of activities are also fixed (Davis [1], Davis and Patterson [2], Moder et al. [8], Willis [15]). In reality, the durations for many activities are not fixed but are consequences of the intensity of resource applications. In particular, work performed in production shops often has this characteristic. For example, in the case of ship overhaul, suppose an activity to repair 10 valves in a shop can be assigned anywhere from 8 to 40 manhours per day and requires 80 total manhours to complete. If the shop capacity currently available for allocation to this job is 10 manhours per day and will become 20 manhours two days later, the shop manager would prefer to allocate 10 manhours to the job for the first two days and then upgrade the allocation to 20 manhours on the third day in order to finish the job in 5 days. Unfortunately, traditional CPM models preclude such flexibility in resource allocations, generating schedules which utilize resources in a less efficient way.

A more generalized production model is necessary to reflect the situations where each activity can be assigned any level within a certain range of resource utilization rates. In such a model, any level of resource rates between some lower and upper feasible bounds can be applied at an activity, and the resource application level can be changed at any time during the execution of the activity. The duration of an activity should be endogenous, a consequence of resource al-

Received January 1988; revised June 1988. Handled by the Scheduling/Planning/Control Department.

 0740-817X/90/$3.00x.00

location decisions and the total resource requirements of the activity.

As a step in this direction, several authors have considered the case where there are discrete alternatives for activity resource loading and duration. Quite some time ago, Wiest [14] introduced a model in which each job has discrete alternatives of normal, maximum and minimum crew sizes with associated durations. For this same type of model, Talbot [10] developed an optimal implicit enumeration procedure.

Several attempts have been made to formulate and solve more general project scheduling problems with continuously flexible activity resource loading. For the resource levelling problem, Leachman [6] developed a heuristic algorithm to solve a model incorporating activity intensities continuously variable between upper and lower bounds. For project scheduling constrained by fixed resource availabilities, Weglarz [11], [12], [13] has proposed single-resource preemptive models in which the activity duration-resource usage relationship is described by a performance speed-resource usage function. His algorithms to find minimum project completion time are based upon the prespecification of a complete ordering of the partially ordered nodes in the project network. Unfortunately, finding an overall optimal schedule for a large project is an impractical combinatoric task, as one needs to compare the results for all possible complete node orderings.

In this paper, we propose heuristic resource allocation algorithms for scheduling a project constrained by limited resources which are assumed to be non-storable and infinitely-divisible such as labor trade services, machine services, etc. The objective of the problem is to find a schedule which minimizes project completion time. In the next section, we set forth our production model for a project-oriented production system. The model was first introduced in Leachman [6]; we adapt it here for the resource-constrained scheduling problem. Later we present two heuristic resource allocation algorithms, which we term Upgrading-Only and Upgrading-and-Downgrading, respectively. Then, we compare the performance of our allocation algorithms to that of traditional fixed-duration scheduling methods in solving 110 sample problems from Patterson [9]. Finally, we suggest some extensions to the production model and the proposed heuristics.

A Model of Project-Oriented Production with Continuously-Variable Intensities

We will consider a project characterized by an activity-on-node network consisting of N activities $A_1, A_2, \ldots, A_N$. The project is assumed to require the services of K infinitely-divisible, non-storable resources. Our basic assumption is that all the different types of resources which are required by an activity are applied proportionally throughout the activity execution, i.e., each activity utilizes a constant mix of resources as it progresses exactly proportional to the mix of total resource requirements to complete the activity. Thus the rates of applications of different types of resources to an activity can be indexed in terms of one rate which will be called the *intensity* of the activity. The activity intensity is assumed to be continuously variable within given upper and lower limits. The rate of progress of an activity is assumed to be proportional to its intensity. In the terminology of Weglarz [13], we assume linear performance speed-resource usage functions; in terms of production theory (Hackman and Leachman [4]), we assume Leontief activity production functions.

Let $z_i(t)$ denote the intensity of activity A_i at time t, i.e., the index of the rates of resource applications to the activity at time t. Let

$\underline{z}_i$ = lower bound on $z_i(t)$,

$\bar{z}_i$ = upper bound on $z_i(t)$,

a_{ki} = total amount of resource k required by A_i.

Then $a_{ki}z_i(t)$ corresponds to the rate resource k is applied to A_i at time t. Let S_i and F_i denote the start and finish times of A_i, respectively. We require that

$$\underline{z}_i \le z_i(t) \le \bar{z}_i \quad \text{for } t \in (S_i, F_i] \tag{1}$$

$$z_i(t) = 0. \quad \text{for } t \notin (S_i, F_i] \,. \tag{2}$$

We assume $z_i(t)$ is normalized so that

$$\int_{S_i}^{F_i} z_i(t)dt = 1.0 \,. \tag{3}$$

We let $Z_i(t)$ denote the *cumulative intensity* of A_i by time t, i.e.,

$$Z_i(t) = \int_0^t z_i(t)dt \,.$$

Note that $Z_i(t)$ is equivalent to the fraction of A_i completed by time t.

In the algorithms presented below, it is necessary on occasion to project the remaining duration of an activity in progress. For $t \in (S_i, F_i]$, if we assume that the current intensity $z_i(t)$ will be maintained on the interval $(t, F_i]$, then the remaining duration of A_i is calculated as

$$\tilde{d}_i(t) = \frac{1.0 - Z_i(t)}{z_i(t)} \,.$$

We assume that each activity has a *normal duration* which corresponds to continuous operation from activity start to activity finish at *normal intensity*. Normal intensity indexes

rates of resource utilization which represent a mode of regular operation for the activity. Such normal durations will be used to estimate target finish times of unscheduled activities in the algorithms explained in the next section.

Heuristic Algorithms for Intensity Assignment

Instead of simply determining start times of activities, in our more general formulation one decides the rates of resource applications to activities, i.e., one assigns intensity levels to the activities. We present two heuristic algorithms, Upgrading-Only and Upgrading-and-Downgrading. In the Upgrading-Only case, it is assumed that management does not allow reduction in the level of resources already committed to an activity, but it allows increases in the level during the activity execution. The Upgrading-and-Downgrading algorithm is based on the assumption that management allows both increases and decreases in the allocated resource levels while the activity is in progress.

To establish priority among active and ready activities, our algorithms utilize the familiar Minimum Slack scheduling rule with parallel updating of activity priorities widely proposed for fixed-intensity scheduling. (See, for example, Davis and Patterson [2].) However, we incorporate additional logic to actually select activity intensity levels, as summarized below.

Each intensity assignment algorithm can be visualized as a decision-maker which decides the level of the intensities of activities which will be maintained until the next decision time. Intensities will be revised at all time points when there is a change in the system that provides incentive or necessity for resource allocation. Such changes include the completion of an activity which frees resources and allows its followers to start, and changes in the levels of the available resources in the case of time-varying resource capacities. At each decision time, the intensities of currently active activities and of newly schedulable ones are revised using a decision process that takes account of their lateness with respect to target finish times.

Initially, the target finish times are taken as the latest finish times of activities as computed with the usual critical path calculations assuming each activity will be performed at normal intensity. Once a certain level of intensity is assigned to an activity, we project its remaining duration by assuming that the current intensity level will be maintained until completion of the activity. Durations for uninitiated activities are still based on normal intensity. At all decision times after time 0, the target (latest) finish times are updated by performing critical path calculations using these revised projections for (remaining) activity durations.

Following the calculation of target finish times, a list is generated of the activities which are currently active and not completed plus those ready to be started. Activities on the list are prioritized on the basis of projected lateness with respect to target finish times. According to these priorities, the activities are considered for intensity assignment one by one. Firstly, all the currently active activities are tentatively assigned minimum-feasible intensity levels, i.e., the lower bound intensity levels (Upgrading-and-Downgrading case) or the same intensity levels as the ones previously assigned (Upgrading-Only case).

Now, any one activity which is late with respect to its target finish time may increase the project completion time, even if all the others finish early. Thus, a sensible criteria at each decision time is to schedule all activities to meet their target finish times, if possible. Hence in the second step, we proceed down the priority list, allocating just enough additional resources (if required and if available), so that all activities would be completed at their target finish times. In this way, we try to equalize the relative lateness of the current activities as much as possible. But if the intensity level corresponding to completion at the target finish time is less than the lower intensity bound, and enough resources are available, we set the intensity at the lower bound in order to schedule the activity at its slowest mode rather than delay it.

After all the activities in the list are considered, if resources are still left over, those resources are allocated to the activities by upgrading the current intensity levels even further. Thus, in the third step, a second pass through the priority list is made in which activity intensities are increased as much as possible, considering the remaining resource levels and intensity bounds.

We now formally describe the detailed steps for the two heuristic algorithms:

Notation

In addition to the previous notation, let

z_i^0 = intensity level of A_i assigned at the last decision time.

d_i = normal duration of A_i.

$TF_i(t)$ = target latest finish time of A_i computed at time t.

$lsc_i(t)$ = lateness score of A_i calculated at time t.

$R_k(t)$ = capacity of resource k at time t. We assume R_k is a step function whose level changes at a finite number of points in time.

$AR_k(t)$ = amount of resource k unallocated at time t.

$ACTIV$ = set of indices of activities which have been started but not yet completed by time t.

$READY$ = set of indices of activities not yet started but all of whose predecessors are completed by time t.

P_i = set of indices of the immediate predecessor ac-

tivities of activity A_i.

Upgrading-Only Algorithm
Initialization.

Set $t = 0$, $z_i^0 = 0$, $Z_i(t) = 0$ for all i,

$AR_k(t) = R_k(t)$ for all k,

$ACTIV = \varnothing$.

Step 1.

(a) Set $READY \leftarrow \{i \mid Z_i(t) = 0.0, Z_j(t) = 1.0 \text{ for all } j \in P_i\}$.

(b) Set $TF_i(t)$ = latest finish time of A_i obtained by the usual critical path calculations assuming the (remaining) duration of A_i is

$$\tilde{d}_i(t) = \begin{cases} d_i & \text{if } Z_i(t) = 0 \\ \dfrac{1.0 - Z_i(t)}{z_i^0} & \text{if } 0 < Z_i(t) < 1.0 \text{ and } z_i^0 > 0 \\ (1.0 - Z_i(t))d_i & \text{if } 0 < Z_i(t) < 1.0 \text{ and } z_i^0 = 0 \\ 0 & \text{if } Z_i(t) = 1.0 \end{cases} \tag{4}$$

(c) Prioritize activities in $ACTIV \cup READY$ in decreasing order of lateness scores calculated as follows:

$$lsc_i(t) = t + \tilde{d}_i(t) - TF_i(t)\,, \tag{5}$$

where $\tilde{d}_i(t)$ is defined by (4).

(d) Set $AR_k(t) \leftarrow AR_k(t) - \sum_{i \in ACTIV} a_{ki} z_i^0$ for all k.

If for some k $AR_k(t) < 0$, then the new (lower) capacity level requires reduction of the intensities of active activities. In this case, we apply an Intensity Adjustment Routine described at the end of this section. After adjusting intensities, go back to Step 1(b). Otherwise, continue to Step 2.

Step 2. Intensity Assignment:

(a) Calculate intensities of activities in $ACTIV \cup READY$ one by one in order of priority as follows. For $i \in ACTIV \cup READY$, we define

$$y_i(t) = \text{Min} \begin{cases} \text{Max}\left\{\dfrac{1.0 - Z_i(t)}{TF_i(t) - t}, \underline{z}_i\right\} \\ \underset{\{k \mid a_{ki} \neq 0\}}{\text{Min}} \left(\dfrac{AR_k(t)}{a_{ki}}\right) + z_i^0 \\ \bar{z}_i \end{cases} \tag{6}$$

The top term of the outer minimization in (6) expresses the rate A_i should be operated at to just complete by the target finish time $TF_i(t)$, but bearing in mind that A_i can not be operated more slowly than $\underline{z}_i$. The middle term of the outer minimization expresses the maximum rate A_i may be operated at considering the available resources, while the bottom term expresses the maximum intrinsic rate $\bar{z}_i$ for A_i. Hence $y_i(t)$ expresses the rate of operation of A_i which yields a finish time for A_i as close as possible to $TF_i(t)$ considering the available resources and the intensity bounds.

We reset the intensity of A_i as

$$z_i(t) = \begin{cases} y_i(t) & \text{if } y_i(t) \geq \text{Max}\{z_i^0, \underline{z}_i\} \\ z_i^0 & \text{if } \underline{z}_i \leq y_i(t) < z_i^0 \\ 0 & \text{if } z_i^0 = 0 \text{ and } y_i(t) < \underline{z}_i \end{cases} \tag{7}$$

The top term in (7) applies if A_i is late or on time with respect to its target finish time; the middle term applies when A_i is active but early; and the bottom term applies when A_i is ready or disrupted but available resources are inadequate to operate A_i at its lower bound.

(b) Update $AR_k(t) \leftarrow AR_k(t) - a_{ki}(z_i(t) - z_i^0)$ for all k.

(c) Repeat Steps 2(a)-(b) until all the activities in $ACTIV \cup READY$ are considered or until $AR_k(t) = 0$ for all k.

Step 3. Use Remaining Resources to Upgrade Intensities:

(a) Upgrade activity intensities as follows. Again proceeding down the list of activities active or ready in order of priority, define

$$w_i(t) = \text{Min} \begin{cases} \bar{z}_i - z_i(t) \\ \underset{\{k \mid a_{ki} \neq 0\}}{\text{Min}} \left(\dfrac{AR_k(t)}{a_{ki}}\right) \end{cases} \tag{8}$$

Reset $z_i(t) \leftarrow z_i(t) + w_i(t)$.

If $z_i(t) < \underline{z}_i$, set $w_i(t) = 0$ and $z_i(t) = 0$.

(b) Update $AR_k(t) \leftarrow AR_k(t) - a_{ki}w_i(t)$ for all k.

(c) Repeat Steps 3(a) & (b) until $AR_k(t) = 0$ for all k or until the list of activities in $ACTIV \cup READY$ is exhausted.

Step 4. Make Necessary Updates:

(a) For all $i \in READY$, if A_i is newly scheduled, update

$$ACTIV \leftarrow ACTIV \cup \{i\}.$$

(b) Find next decision time:

$$t_{next} = \text{Min} \begin{cases} \underset{i \in ACTIV}{\text{Min}} \left\{ t + \dfrac{1.0 - Z_i(t)}{z_i(t)} \right\} \\ \underset{k}{\text{Min}} \underset{\tau > t}{\text{Min}} \{\tau | R_k(\tau) \neq R_k(t)\} \end{cases} \tag{9}$$

(c) For all $i \in ACTIV$, if t_{next} is the completion time of A_i (i.e., $t + d_i(t) = t_{next}$), then reset

$$ACTIV \leftarrow ACTIV - \{i\},\ Z_i(t_{next}) = 1.0.$$

(d) Set $Z_i(t_{next}) = Z_i(t) + (t_{next} - t)z_i(t)$, and

$$z_i^0 = z_i(t) \text{ for all } i \in ACTIV.$$

Reset $t \leftarrow t_{next}$, and

$$AR_k(t) \leftarrow R_k(t) \text{ for all } k.$$

Step 5. Stop if all the activities are completed. Otherwise, return to Step 1 to process the next decision time.

Upgrading-and-Downgrading Algorithm

The Upgrading-and-Downgrading algorithm is the same as the Upgrading-Only algorithm, except one sets $z_i^0 = \underline{z}_i$ in Step 4(d) of the Upgrading-Only algorithm. Thus, all the active activities are temporarily assigned their lower intensity bounds, and then upgraded according to priority using the Upgrading-Only algorithm.

Intensity Adjustment Routine

When the resource capacities are time-varying, we may encounter a situation in which the current intensity levels are not feasible under a new lower capacity. In that case, we need to downgrade intensities of the currently active activities. In the following steps, we lower the intensity levels of the activities one by one following the reverse order of priorities set by the lateness scores in Step 1(c) of the intensity assignment algorithms.

Step 1.

(a) Calculate the amount of resources in excess of capacity needed to maintain the current intensity levels of the active activities. For all k, define

$$r_k = \text{Max } \{-AR_k(t), 0\}. \tag{10}$$

(b) Select the active activity A_i following the reverse priority order, and set

$$x_i = \text{Max} \begin{cases} z_i^0 - \underset{\{k | a_{ki} \neq 0\}}{\text{Max}} \left(\dfrac{r_k}{a_{ki}} \right) \\ \underline{z}_i \end{cases} \tag{11}$$

(c) Update $AR_k(t) \leftarrow AR_k(t) + a_{ki}(z_i^0 - x_i)$.

Set $z_i^0 = x_i$.

(d) If $AR_k(t) \geq 0$ for all k, return to Step 1(b) of the Upgrading-Only algorithm. Otherwise, go back to Step 1(a) of this routine to downgrade another activity intensity. If all active activities have been processed, continue to next step.

Step 2.

In this step, the active activities are disrupted until capacity violations are eliminated. Activities are disrupted one by one according to the reverse order of priorities.

(a) Recalculate activity priorities by performing Steps 1(b) and 1(c) of the Upgrading-Only algorithm using the modified z_i^0's computed in Step 1 above.

(b) Select an active activity A_i following the reverse priority order. If $a_{ki} \neq 0$ for some k such that $AR_k(t) < 0$, then set

$$AR_k(t) \leftarrow AR_k(t) + a_{ki}z_i^0, \text{ and } z_i^0 = 0.$$

(c) If $AR_k(t) \geq 0$ for all k, return to Step 1(b) of the Upgrading-Only algorithm. Otherwise, go back to Step 2(b) of this routine to disrupt the next active activity in reverse order of priority.

Computational Results

Important questions about our intensity-based approach to scheduling which must be answered computationally are as follows:

1. How much can project duration be shortened by using our intensity assignment algorithms in lieu of traditional fixed intensity CPM methods?

2. How important is it to carry out a two-phase allocation, i.e., to first allocate just enough resources to all activities so that target finish times can be achieved, and to then allocate remaining resources in a greedy fashion in order of minimum slack priority? Wouldn't a single greedy allocation in order of minimum slack priority suffice?

3. Are the intensity assignment algorithms computationally practical for large-scale project scheduling?

To answer these questions, the algorithms introduced above have been coded in FORTRAN and have been tested extensively on an IBM 3091 computer. 110 test problems were contributed by James H. Patterson that were originally used for comparing performances of optimal procedures for the fixed-intensity case (Patterson [9]). To test our variable-intensity scheduling algorithms, we added an assumption that each activity that uses resources can be assigned intensity levels within some given range of normal intensity, where normal intensity of an activity is taken as the reciprocal of the given duration for the activity. For example, assuming a range of .8-1.2, an activity can be assigned any intensity level between .8/(given duration) and 1.2/(given duration). If an activity in a test problem uses no resources but has positive duration, its duration was assumed to be fixed. For description of the test problems, we refer the reader to Patterson [9].

To address the first question, optimal project completion times for the fixed-intensity cases, as well as completion times generated using the usual fixed-intensity Minimum Slack rule with parallel updating, were compared with completion times generated by our intensity assignment algorithms for the 110 test problems. To address the second question, single-pass greedy assignment algorithms, as suggested in question 2. above, were also tested. (An implementation of the single-pass algorithms was obtained simply by skipping Steps 2(d) and 3 of our intensity assignment algorithms.) Two allowed ranges of normal intensity, 0.8-1.2 and 0.5-1.5, were tested for both the Upgrading-Only and the Upgrading-and-Downgrading intensity assignment algorithms and for the corresponding single-pass modifications as well.

The results are summarized in Table 1. Average, minimum, maximum and standard deviation of the percentage decrease in project completion times over the fixed-intensity project completion times are reported. Depending on the range allowed for intensity assignment and on whether upgrading-only or upgrading-and-downgrading is allowed, our heuristic intensity assignment algorithms outperform *optimal* fixed-intensity schedules by about 6-14%, and they outperform heuristic fixed-intensity schedules by about 10-18%. About one fifth to one half of this improvement (i.e., about 3% in every case) is attributable to the use of our two-phase allocation procedure in lieu of a single-pass greedy assignment. Occasionally, the intensity assignment algorithms performed inferior to fixed-intensity methods, but this behavior we attribute to the combinatoric nature of small, significantly serial networks. In our experience, large industrial project networks have a highly parallel structure, and we conjecture that our assignment algorithms perform consistently well on such networks.

To address the third question, our intensity assignment algorithms have been utilized to schedule (on a demonstration basis) naval ship overhaul projects involving upwards of 3000 activities constrained by more than 50 scarce services. Total CPU time on an IBM 3091 to schedule such a project is less than 20 seconds.

The conclusion emerging from these results is that, if there is flexibility possible in activity resource loading, it should be modelled in scheduling procedures. The wider the feasible range of activity intensity, the less constrained is the scheduling problem and the greater is the opportunity for efficient resource utilization. Our heuristic procedures for intensity assignment are computationally practical, and the schedules they generate promise significant resource pro-

Table 1. Summary of the Performance of Intensity Assignment Algorithms vs. Fixed-Intensity Scheduling Methods

		Allowed Intensity Range	Project Completion Times							
			% Under Fixed-Intensity Heuristic				% Under Fixed-Intensity Optimum			
			Average	Max.	Min.	Std. Dev.	Average	Max.	Min.	Std. Dev.
Proposed Intensity Assignment Algorithms	Upgrading-Only	0.8-1.2	10.4	23.6	0.0	4.5	5.8	16.7	−13.7	5.6
		0.5-1.5	17.5	30.4	−1.3	5.8	13.3	30.4	−3.7	6.1
	Upgrading-and-Downgrading	0.8-1.2	12.6	25.2	−3.1	4.9	8.1	20.0	−14.7	5.6
		0.5-1.5	18.0	30.4	−2.5	5.5	13.8	30.4	−3.7	6.0
Single Pass, Greedy Assignment Algorithms	Upgrading-Only	0.8-1.2	7.2	21.1	−6.3	5.2	2.5	15.7	−14.7	6.4
		0.5-1.5	14.5	29.4	−5.2	6.1	10.1	29.4	−10.5	7.0
	Upgrading-and-Downgrading	0.8-1.2	9.7	22.6	−2.6	5.0	5.1	16.7	−10.5	5.4
		0.5-1.5	15.3	30.7	−0.5	5.7	11.0	30.7	−5.5	6.7

ductivity improvements over even optimal fixed-intensity procedures.

Extensions

Production-Like Workflow Dependencies

In conventional critical path analysis, activities are related by strict event-based precedence, where each successor cannot start until all of its predecessors are finished. Occasionally, in order to reduce the number of activities, or by nature of the operations, overlap relationships are useful modelling techniques. For example, suppose we are given two activities, repairing 10 valves and reinstalling 10 valves. The former precedes the latter, with the understanding that after each valve is repaired, the same valve may be reinstalled. To precisely characterize workflow dependencies in terms of conventional, strict-precedence CPM activities would require the definition of 10 valve repair activities and 10 valve reinstallation activities, for a total of 20 activities. However, the measurement of cumulative activity intensities provides the opportunity to enforce a reasonably accurate inventory balance relationship between the single (aggregate) valve repair activity and the single (aggregate) valve reinstallation activity in lieu of strict precedence relationships among many detailed activities. Such an inventory balance relationship enforces the requirement that the cumulative intensity of the valve reinstallation activity A_i must lag behind the cumulative intensity of the valve repair activity A_j by at least 10% until the repair activity A_j is completed. In such a case we say there is a *progress lag* $f_{ji} = 0.10$ between A_j and A_i, and that activities A_j and A_i are related by a *flow transfer*. In this section, we extend the model of production and the intensity assignment algorithms to accommodate flow transfers.

The model of production is modified as follows. Given a progress lag f_{ji} between A_j and A_i, where $0 < f_{ji} \leq 1$, the requirement of strict precedence between A_j and A_i is replaced by the following constraint on (cumulative) activity intensities:

$$Z_i(t) \leq \begin{cases} 0.0 & \text{if } 0 \leq Z_j(t) < f_{ji} \\ Z_j(t) - f_{ji} & \text{if } f_{ji} \leq Z_j(t) < 1.0 \\ 1.0 & \text{if } Z_j(t) = 1.0 \end{cases} \qquad (12)$$

Note that A_i cannot start until A_j is $100f_{ji}\%$ complete, and until A_j is fully completed, the progress of A_i must lag $100f_{ji}\%$ behind the progress of A_j. Of course, $f_{ji} = 1$ corresponds to strict precedence. Interpreting the flow transfer model, we may view A_j as producing intermediate product required as input by A_i. After an initial (quantity) lag, A_j is assumed to supply an infinitely divisible intermediate product to A_i, allowing progress to be made towards completion of A_i. In practice, we have found that the use of flow transfers significantly reduces the time and effort to prepare activity networks for large industrial projects.

With flow transfer relationships, two activities A_j and A_i related by a flow transfer both will be active at certain points in time. Given that (12) is currently satisfied at such a time point, new intensity assignments are to be made insuring that (12) will be satisfied continuously until the next reallocation event. It is easily verified that if (12) is satisfied at two points in time when both A_j and A_i are active, and if intensities are held constant in between those two time points, then (12) is satisfied at all time points in between. Hence we can guarantee inventory balance between the current time t and the next reallocation event if we establish activity intensities $z_j(t)$ and $z_i(t)$ so as to insure inventory balance at the time A_j would be finished. That is, given that we currently have inventory balance, i.e.,

$$Z_i(t) \leq Z_j(t) - f_{ji},$$

we wish to select $z_j(t)$ and $z_i(t)$ such that

$$\frac{1.0 - Z_j(t)}{z_j(t)}[z_i(t)] + Z_i(t) \leq 1.0 - f_{ji},$$

or

$$z_i(t) \leq \frac{z_j(t)\,[1.0 - f_{ji} - Z_i(t)]}{1.0 - Z_j(t)}. \qquad (13)$$

In the modified intensity assignment algorithms presented below, the predecessor A_j always receives an intensity assignment before the follower A_j, with the assignment to A_j constrained by (13). Formally, the procedures of the intensity assignment algorithms are modified as follows to accommodate flow transfers:

Notation: same as before, with the following additions:

Λ_i = set of indices of activities which immediately follow A_i, including those which have a flow transfer relationship with A_i,

Γ_i = set of indices of activities which immediately precede A_i, including those which have a flow transfer relationship with A_i,

$FT_{il}(t)$ = the flow transfer time of A_i for its immediate follower A_l, i.e., the time that A_i will be progressed enough to allow A_l to start, projected at time t.

Step 1.

(a) Same as before, except define the set

$$READY \leftarrow \{i | Z_i(t) = 0.0,\ Z_j(t) \geq f_{ji} \text{ for all } j \in \Gamma_i\}$$

(b) Estimate the remaining duration of each activity A_i as in equation (4). Compute $TF_i(t)$ = latest finish time of A_i using modified critical path calculations explained

as follows. In the forward pass, set

$$ES_i(t) = \begin{cases} t & \text{if } \Gamma_i = \varnothing \\ & i=1,2,\ldots,N, \\ \underset{j\in\Gamma_i}{\text{Max}}\{ES_j(t) + d^f_{ji}(t)\} & \text{otherwise} \end{cases} \tag{14}$$

where

$$d^f_{ji}(t) = \begin{cases} f_{ji}d_j & \text{if } Z_j(t) = 0 \\ \dfrac{f_{ji} - Z_j(t)}{z^0_j} & \text{if } 0 < Z_j(t) < f_{ji} \text{ and } z^0_j > 0 \\ [f_{ji} - Z_j(t)]d_j & \text{if } 0 < Z_j(t) < f_{ji} \text{ and } z^0_j = 0 \\ 0 & \text{if } Z_j(t) \geq f_{ji}\ . \end{cases} \tag{15}$$

Let DD be the due date for the project.[1] In the backward pass, set

$$TF_i(t) = \begin{cases} DD & \text{if } \Lambda_i = \varnothing \\ & i=N,N-1,\ldots,1, \\ \underset{j\in\Lambda_i}{\text{Min}}\{TF_j(t) - d^b_{ij}(t)\} & \text{otherwise} \end{cases} \tag{16}$$

where

$$d^b_{ij}(t) = \begin{cases} \dfrac{f_{ij}}{z^0_j} & \text{if } Z_j(t) \leq 1.0 - f_{ij} \text{ and } z^0_j > 0 \\ f_{ij}d_j & \text{if } Z_j(t) \leq 1.0 - f_{ij} \text{ and } z^0_j = 0 \\ \tilde{d}_j(t) & \text{otherwise}\ . \end{cases} \tag{17}$$

(c) Same as before, with the following addition: In the case that the lateness scores of two activities with a flow transfer relationship are the same, insure that the predecessor is given a higher priority ranking than the follower.

(d) Same as before.

Step 2.

(a) Same as before, with the following addition at the end of step (a): If $\Gamma_i \neq \varnothing$, define

$$z'_i(t) = \text{Min}\left\{z_i(t), \underset{\{j|j\in\Gamma_i \text{ and } Z_j(t)<1.0\}}{\text{Min}} \frac{z_j(t)[1.0 - f_{ji} - Z_i(t)]}{1.0 - Z_j(t)}\right\}, \tag{18}$$

and reset

$$z_i(t) = \begin{cases} z'_i(t) & \text{if } z'_i(t) \geq \underline{z}_i \\ 0 & \text{otherwise}\ . \end{cases} \tag{19}$$

The rest of Step 2 is the same as before.

Step 3.

(a) Same as before, except, if $\Gamma_i \neq \varnothing$, include the expression

$$\underset{\{j|j\in\Gamma_i \text{ and } Z_j(t)<1.0\}}{\text{Min}} \left\{\frac{z_j(t)[1.0 - f_{ji} - Z_i(t)]}{1.0 - Z_j(t)} - z_i(t)\right\} \tag{20}$$

in taking the minimum in (3.5) defining $w_i(t)$.

The rest of Step 3 is the same as before.

Step 4.

Add the following step between Steps 4(a) and 4(b):

(a′) Update flow transfer times of upgraded activities: Set

$$FT_{il}(t) = t + \frac{f_{il} - Z_i(t)}{z_i(t)} \text{ for all } l\in\Lambda_i \text{ such that } Z_i(t) < f_{il}.$$

Modify Step 4(b) as follows:

(b) Find next decision time:

$$t_{next} = \text{Min} \begin{array}{l} \underset{i\in ACTIV}{\text{Min}}\ t + \dfrac{1.0 - Z_i(t)}{z_i(t)}, \underset{\{l|l\in\Lambda_i \text{ and } Z_i(t)<f_{il}\}}{\text{Min}} FT_{il}(t) \\ \underset{k}{\text{Min}}\ \underset{\tau>t}{\text{Min}}\{\tau | R_k(\tau) \neq R_k(t)\} \end{array} \tag{21}$$

The rest of Step 4 is the same as before.

Step 5.

Same as before.

[1] If there is no exogenous due date for the project, DD can be set equal to the early finish time for the project as calculated in the forward pass.

Hierarchical Linkage with Aggregate Planning of Multiple Projects

Aggregate planning models for multi-project resource allocation (Leachman and Boysen [7], Hackman and Leachman [5]) establish both resource allocations and milestone due dates for individual projects. The allocations serve as resource capacities for resource-constrained scheduling of individual projects, with the additional proviso that milestone dates must be observed (if possible). Milestone dates represent exogenous due dates or exogenous constraints on start dates for various activities of a project. For example, in a naval ship overhaul, milestone dates might be established for such events as docking, undocking, nuclear plant criticality, etc., with upwards of 100 due dates established for a large project. Observance of both the milestone dates and the resource allocations is necessary to realize resource usage consistent with inter-project planning.

To accommodate milestone due dates, the only modifications to the intensity assignment algorithms involve the inclusion of any exogenous constraints on start dates in the CPM forward pass, and the inclusion of the exogenous due dates in the comparisons of the CPM backward pass. (These CPM calculations arise in the calculation of target finish times.) To handle exogenous constraints on activity start dates, dummy activities requiring no scarce resources but with the appropriate fixed durations can be added to the activity network.

Strategic Disruption

The Upgrading-and-Downgrading algorithm presented here avoids activity disruption (i.e., downgrading activity intensity to 0) unless $\underline{z}_i = 0$. In a situation where activities with nonzero minimum intensities may be disrupted without penalty, it would be advantageous to modify the algorithm to consider strategic disruption in making intensity assignments. Specifically, in Step 2 of the algorithm, activity intensity could be set to 0 if operation at or above the lower intensity bound is not required to meet the target finish time.

Mixed Upgrading-Only and Upgrading-and-Downgrading

In an industrial project network, field-type activities might be classified as Upgrading-Only activities while shop-type activities might be classified as Upgrading-and-Downgrading activities. The Upgrading-and-Downgrading algorithm is modified in the obvious way to handle this case: One sets $z_i^0 = \underline{z}_i$ in Step 4(d) for only those activities A_i which are Upgrading-and-Downgrading activities.

Acknowledgements

This research was partially supported by the Office of Naval Research and the Puget Sound Naval Shipyard under contract N00014-76-C-0134 with the University of California at Berkeley. The cooperation of the staff of the Puget Sound Naval Shipyard is gratefully acknowledged. Portions of this paper were adapted from Dr. Dincerler's dissertation, Dincerler [3]. The authors are indebted to the late Professor Ronald W. Shephard, who first proposed the use of Leontief production functions to model project activities and the development of heuristic intensity assignment algorithms. Our thanks also go to Professor James H. Patterson for providing us with the data for the test problems.

REFERENCES

[1] Davis, E. W., "Project Scheduling under Resource Constraints —Historical Review and Categorization of Procedures," *AIIE Transactions 5*, 297-313 (1973).

[2] Davis, E. W. and J. H. Patterson, "A Comparison of Heuristic and Optimum Solutions in Resource-Constrained Project Scheduling," *Management Science 21* (8), 944-955 (1975).

[3] Dincerler, A., "Project Scheduling in Project-Oriented Production Systems," ORC Report 85-11, Engineering Systems Research Center, University of California, Berkeley (1985).

[4] Hackman, S. T. and R. C. Leachman, "A General Framework for Modeling Production," *Management Science 35* (4), 478-495 (1989).

[5] Hackman, S. T. and R. C. Leachman, "An Aggregate Model of Project-Oriented Production," *IEEE Transactions on Systems, Man and Cybernetics* 19 (2), 220-231 (1989).

[6] Leachman, R. C., "Multiple Resource Leveling in Construction Systems Through Variation of Activity Intensities," *Naval Research Logistics Quarterly 30* (3), 187-198 (1983).

[7] Leachman, R. C. and J. Boysen, "An Aggregate Model for Multi-Project Resource Allocation," in *Project Management: Methods and Studies*, B. V. Dean, ed., North Holland, Amsterdam (1985).

[8] Moder, J. J., C. R. Phillips and E. W. Davis, *Project Management with CPM and PERT*, 3rd Edition, Van Nostrand Reinhold, New York (1985).

[9] Patterson, J. H., "A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem," *Management Science 30* (7), 854-867 (1984).

[10] Talbot, F. B., "Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: The Nonpreemptive Case," *Management Science 28* (10), 1197-1210 (1982).

[11] Weglarz, J., "Time-Optimal Control of Resource Allocation in a Complex of Operations Framework," *IEEE Trans. Systems, Man and Cybernetics 6*, 783-788 (1976).

[12] Weglarz, J., "Project Scheduling with Discrete and Continuous Resources," *IEEE Trans. Systems, Man and Cybernetics 9*, 644-650 (1979).

[13] Weglarz, J., "Project Scheduling with Continuously-Divisible, Doubly Constrained Resources," *Management Science 27* (9), 1040-1053 (1981).

[14] Wiest, J. D., "A Heuristic Model for Scheduling Large Projects with Limited Resources," *Management Science 13* (6), 1120-1148 (1967).

[15] Willis, R. J., "Critical Path Analysis and Resource Constrained Project Scheduling — Theory and Practice," *European Journal of Operational Research 21*, 149-155 (1985).

Robert C. Leachman is Associate Professor of Industrial Engineering and Operations Research at the University of California at Berkeley. He received the A.B. degree in Mathematics and Physics and the M.S. and Ph.D. degrees in Operations Research, all from U.C. Berkeley. He received the Nicholson Prize from the Operations Research Society of America in 1980. Dr. Leachman's areas of research include production planning and scheduling systems and transportation planning. He is a member of IIE, TIMS, ORSA and APICS.

Abdurrezak Dincerler is currently Vice President of Muhas Inc. in Istanbul, Turkey. He received the B.S. degree in Industrial Engineering from the Middle East Technical University and the M.S. and D. Eng. degrees in Industrial Engineering from the University of California at Berkeley.

SooYoung Kim is Assistant Professor in the Dept. of Industrial Engineering at Rutgers University. He received the B.S. degree in Mechanical Engineering from Seoul National University, the M.S. degree in Manufacturing Engineering from Korea Advanced Institute of Science and Technology, and the Ph.D. degree in Industrial Engineering from the University of California at Berkeley. Mr. Kim is a member of IIE and ORSA.