A Multi-Resource Leveling Algorithm for Project Networks[*]

Chung Ung Lee[**]

ABSTRACT

This thesis presents a modification and extension to the Burgess and Killebrew heuristic resource leveling procedure for project networks. In contrast to previous algorithms appea ring in the literature, the objective function of this algorithm is the minimization of the sum of the squared errors in each time period (deviations around the mean usage) of all resources over the duration of the project. This objective function continues the search for an improved schedule beyond that of previous algorithms with their associated objective functions. One important feature is that the algoritm tends to reduce the number of periods that a resource is idle during its duration on the project.

I. INTRODUCTION

A. THE PROBLEM

A resource is a physical quantity such as labor, space, equipment, material or money, the use of which may impose restrictions on the minimum duration of a project. Typically, there are several types of resources which must be considered and each of many activities may require one or more of these resources.

In some project situations, resources can be acquired or released in practically any desired amounts if one is willing to pay the expenses (such as the costs of hiring, training, unemployment insurance, and so on) involved in changing resource levels. It is usually prudent, however, to maintain relatively stable employment levels and to utilize resources at a more uniform rate. For such situations resource leveling programs are most appropriate.

As Moder and Phillips [5] have observed, "resource allocation in project scheduling is probably receiving more attention today than any other aspect of PERT/CPM because modern technology has developed many large and expensive physical resources which must be accounted for and the number of different resources is increasing due to growing specialization and technologies. Personnel resources come in many different trades which are further broken down by skill, geographical location, departmental barriers, etc."

B. A REVIEW OF RECENT SOLUTION METHODS

Many attempts [4] have been made to develop methods for solving the problem of unlimited

[*] Naval postgraduate school에 제출됐던 석사 논문임.
[**] 陸軍本部

resource leveling. Because all of these are heuristic in nature, they do not guarantee optimal solutions. Of particular interest for this thesis are the approaches described below. The scheduling problem addressed in all of these approaches as well as in this thesis is that of resource leveling over minimum project duration when:

1. *the duration times of the activities are fixed, known quantities;*

2. the resource demands by each activity are specified and are assumed to be constant during the duration of the activity;

3. *once processing of an activity begins, it cannot be interrupted.*

The first systematic approach to this problem appears to have been developed by Burgess and Killebrew [1]. Their method of generating and comparing alternative schedules is based on sequentially adjusting starting times of slack activities. The measure of effectiveness they propose for comparison of schedules is the minimization of the sum of the squares of the individual resource requirements in each time period.

Levy, Thompson and Wiest [2] describe a computer program for smoothing manpower requirements which is similar in many respects to the Burgess procedure. In this method, all jobs initially are scheduled to begin at their earliest possible start times, then slack activities scheduled on peak workload days are selected at random and shifted to later days beyond the peak days until further shifting no longer reduces the peak loads. The first segment of the program smoothes the work-loads in all shops simultaneously; the second seqment performs fur.her smoothing on individual shops, beginning with the most expensive shop.

Dewitte [3] describes a computerized manpower leveling procedure developed at Hughes Company Aircraft. His objective function is to minimize the absolute magnitude of fluctuation from a project mean work-load level. Although the procedure also adjusts the start time of projec activities having slack, it breaks up the resource profiles into specially derived intervals and levels within these intervals.

C. THESIS APPROACH TO THE PROBLEM SOLUTION

This thesis will extend the procedure described by Burgess and Killebrew. The steps of the Burgess and Killebrew procedure are presented in the appendix for the convenience of the reader.

The measure of effectiveness which will be used is the minimization of the error sum of squares of all resources over all time periods. This measure incorporates all of the advantages of the measures of effectiveness of the three approaches described above. It is at least as good as any of them and in many problems it is better based on the notion in section A of "maintaining relatively stable employment and utilizing resources at a more uniform rate." The minimization of the error sum of squares (ESS) is going to place more emphasis on large fluctuations around a mean than the Dewitte approach. The Levy, Thompson and Wiest procedure concentrates on reducing the peak workloads only. Once a schedule is reached where further reduction in peak loads early in a project results in an increase above that peak load value later in the project, the procedure terminates. Minimization of the ESS would give comparable results for these peak loads but the loading fluctuations between these peaks would also be smoothed.

The Burgess-Killebrew objective of minimizing the sum of the squares of resources used can be shown to give identical results as those minimizing the ESS if schedules allowing idle periods betweee the start and finish of the usage of a resource on the project are acceptable. However, allowing idle periods is counter to the notions of stable employment and resource utilization at a uniform rate. This thesis considers that it is important to keep these idle periods to as few as possible and the minimization of the ESS may continue the search for better schedules even after the Burgess, Killebrew objective has ceased to be improved.

For comparison of these two objective functions, suppose resource j is used over m_j time periods (including some idle periods), then

$$ESS_j = \sum_{i=1}^{mj} [X_{ij} - \bar{X}_j]^2 = \sum_{i=1}^{mj} X_{ij}^2 - m_j \bar{X}_j^2, \quad (1)$$

where X_{ij} is the amount of resource j used during period i ($i=1$ corresponds to the first period of usage on the project and $i=m_j$ corresponds to the last period of usage), and

$$\bar{X}_j = \frac{\sum_{i=1}^{mj} X_{ij}}{m_j} \quad (2)$$

Now the sum $\sum_{i=1}^{mj} X_{ij}$ is a constant over the project. If m_j is a constant then

$$ESS_j = \sum_{i=1}^{mj} X_{ij}^2 - (\text{a constant})$$

There, using the ESS gives the same results as using the sum of the quantities resources squared, $\sum_{i=1}^{mj} X_{ij}^2$, of Burgess and Killebrew.

Suppose now that several schedules result in the same $\sum_{i=1}^{mj} X_{ij}^2$ but the m_j values differ. That schedule having the smallest m_j value will then give the smallest ESS_j value because of the negative term in equation (1). The m_j term appears only in the denominator of the negative term in (1) after substitution of (2) for $\bar{X}_j$.

The number which will be used for comparing example schedules in this thesis is the total error sum of squares of all n resources; that is,

$$ESS = \sum_{j=1}^{n} ESS_j \quad (3)$$

This expression suggests equal weights among the resources (they are of equal value or comparable cost). An alternative approach is to let

$$ESS = \sum_{j=1}^{n} k_j \, ESS_j \quad (4)$$

where $k_j \leq 0$ is a weight on the jth resource and the ratio k_j/k_i measures the relative importance between resource j and resource i.

In addition to an improved measure of effectiveness, the algorithm presented in this thesis expands the search procedure for improved schedules beyond that of Burgess,Killebrew. As the reader will note in the Appendix, the Burgess-Killebrew approach specifies movement only to the right (beyond their early start times) of activities which are slack. Possible shifts to the left are suggested by Moder and Philips [5] as their interpretation of step 8. The algorithm presented below begins (following Burgess and Killebrew) with activity movement to the right starting with the last activity on the project (step 3). This movement proceeds one time period at a time until the activity's completion date corresponds to its latest finish time based on the critical path or any one of the ESS_j's would increase when the activity was moved one more period to the right. Step 4 continues the movement to the right in the same fashion as Burgess and Killebrew.

Step 5 departs from the Burgess-Killebrew algorithm and initiates movement to the left beginning with the earliest activity having slack at the end of step 4. The second part of step 5 initiates movement again to the right beginning this time with the latest activity having slack.

Step 6 continues the movement to the left and right begun in step 5 but it allows the ESSj to increase on some resources providing that it decreases sufficiently on others to that the sum total over all (ESS) either decreases or remains unchanged.

Ⅱ. THE ALGORITHM

STEP 1.

List project activities in order of precedence as shown on the network diagram by arranging the activity head node numbers in ascending order, and then if two or more activities have arrow the same head node numbers, list themso that the activity tail node numbers are also in ascending order (this assumes that the network activities are numbered so that activity tail node numbers sare always less than head node numbers).

STEP 2.

1. Schedule all activities at their earliest start times and add activity duration (D),earliest start time (ES), total slack (S) and latest start time (LS) for each activity to the listing begun in Step 1.
2. Determine the activity free slack for all activities not on the critical path.
3. Calculate the ESS_j for each resource.

STEP 3

1. Beginning with the last activity having free slack (the nearest the bottom of the list), schedule it to give the lowest ESS_j for each resource. In this Step, the ESS_j's of all resources are minimized simultaneously. Move the activity to the right one period at a time and, for each location, determine the ESS_j· for each j. Continue movement until some ESS_j will increase in the next time period. After finding the optimum location of the last slack activity, adjust the permissible latest finish time (and hence the activity free slack for preceding slack activities) in cases where the last slack activity affects such a finish time.
2. Holding the last slack activity in its new location, repeat Step 3-1 on the next latest slack activity in the list, taking advantage of any slack that may have been made available to it by the scheduling in Step 3-1 of the last slack activity.

3. Continue sequentially scheduling earlier and earlier slack activities from the list according to Step 3-1. Step 3 is finished when *the earliest slack activity has been examined.*

STEP 4.

1. Carry out additional rescheduling cycles using movement to the right until no further reduction in the ESS_j for some resource is possible without increasing the ESS_j of some other resource. If the last slack activity on the list after Step 3 was not moved all the way to the right, then start moving it again to the right and follow the procedure of Step 3 again.
2. *If the last slack activity on the list after* Step 3 has been moved all the way to the right, then find the latest slack activity which can be moved to the right and start moving it according to the Stèp 3 procedure.
3. Repeat Steps 4-1 and 4-2 until no further reduction in the ESS_j is possible for some resource without increasing the ESS_j for some other resource.

STEP 5.

1. *Starting with the first slack activity* on the list in the current schedule, try to schedule it to give a reduction in the ESS_j of at least one resource by moving it to the left one period at a time. Such movement must not increase any other ESS_j. Moving a slack activity to the left is permissible if its current start time is later than its earliest start time.

A. If it cannot be moved to the left or doesn't give an improvement in some ESS_j, then find the earliest slack activity which can be moved to the left and give an improvement in some ESS_j.

B. After moving an activitay to the left, adjust the permissible erlieststart time for affected succeeding activities and the latest finish time for affected preceding activities. Continue movement to the left of successively later slack activities until the list is exhausted.

2. If Step 5-1-A gives an improvement of at least one ESS_j, then, starting with the last slack activity, try to schedule it to give an improvement of at least one ESS_j by moving it to the right one period at a time.

A. If it cannot be moved to the right or does not result in an improvement of some ESS_j, then find the latest slack activity which can be moved to the right and give an improvement in some ESS_j.

B. After moving an activity to the right, adjust the permissible earliest start time for any affected succeeding activity and the latest finish time for any affected preceding activity.

3. Return to Step 5-1 and continue repeating Steps 5-1 and 5-2 until no further reduction in any ESS_j is possible.

STEP 6.

After Step 5, carry out additional rescheduling cycles using the procedures of Step 5 with the modification that reduction in some ESS_j is allowed at the expense of an increase in some other ESS_j provided that the total ESS is reduced. The procedure of Step 5-2 is allowed even if Step 5-1 results in no reduction in the total ESS.

Ⅲ. AN EXAMPLE

The following is a numerical example to illustrate the details of the Chapter II algorithm.

Figure 1 shows the precedence network. The project starts at node 1 and terminates at node 8. There are 3 numbers associated with each arc. These represent, respectively, the duration of the activity and the required numbers of resources A and B. For example, activity (1-3) has a duration of 2 (weeks) and needs 4 of the type A resource and none of the type B.

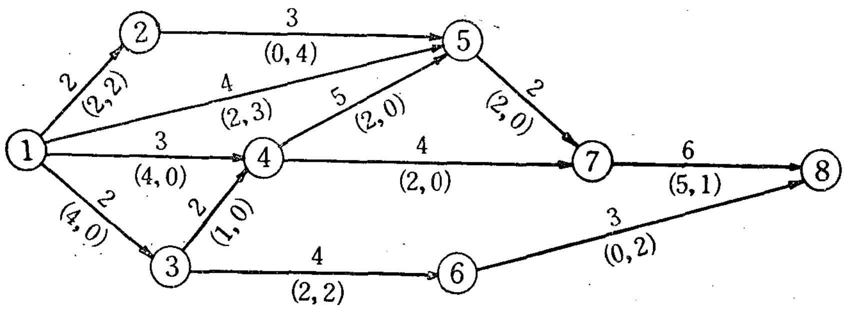

Figure 1

STEP 1.

One important requirement in step 1 is that the activities must be listed in order of precedence as shown in Table 1. By numbering the network nodes so that the node number at the tail of an activity arrow is always less than the node number at the head of an activity arrow, we are able to arrange the activities in order of precedence merely by listing them so that the activity arrow-head numbers are in ascending order. This arrangement of the activities has the property that all of an activity's predecessors will be found above it in the table, and all of its successors will be found below it in the table.

In Table 1, the listing follows the arbitrary additional rule of arranging the activity tail numbers in increasing order when two or more activity arrows have the same head numbers. One could just as easily devise some other rule and obtain a slightly different sequence.

ACTIVITY	RESOURCE		DURATION
	A	B	
1～2	2	2	2
1～3	4	—	2
1～4	4	—	3
3～4	1	—	2
1～5	2	3	4
2～5	—	4	3
4～5	2	—	5
3～6	2	2	4
4～7	2	—	4
5～7	2	—	2
6～8	—	2	3
7～8	5	1	6

Table 1

STEP 2.

All activities are scheduled at their earliest start times (ES) and the ESS_j is calculated for each resource as shown in Figure 2 and all ESS_j value in the example are rounded to the nearest integer. In this example, two resources are involved. In cases where there are several different types that must be considered, then each should be listed;they can be treated sequentially. The bar chart of Figure 2 is prepared in an obvious way from the data given in Table 1. Each activity has been scheduled as early as possible so that it occupies its earliest start and finish time interval. For example, the activity (1-5) begins at the beginning of week no. 1 (period 1), has a duration of 4 weeks and needs 2 of resource type A and 3 of resource B in each of time periods 1, 2, 3, and 4. The "bar" for activity (1-5) is extended by an amount equal to its free slack; this time can be used by the activity without affecting the schedule of any other activity.

The activities on the critical path are surrounded by heavy lined boxes. Because they are on the critical path, they cannot be moved without affecting project duration.

STEP 3.

The latest free slack activity is 6-8 which has 8 weeks of free slack. However, it can only be moved to the right 1 or 2 weeks without increasing the value of ESS_B. Therefore activity 6-8 should be moved 2 weeks to the right.

Next, the activity 4-7 can be moved 3 weeks to the right. A reductions of 16 in the value of ESS_A results.

Activity 3-6 now has 2 weeks of free slack. However, activity 3-6 should be moved only 1 week to the right. Reductions of 20 in the ESS_A and of 28 in the ESS_B result.

Activity 2-5 then has 4 weeks of free slack and can be moved 4 weeks to the right with a reductions of 48 in the ESS_B. Activity 1-5 has 5 weeks of free slack but can be moved only 2 weeks to the right with a reductions of 48 in the ESS_A and without change in the ESS_B (further movement to the right would increase ESS_B).

Finally, activity 1-4 has 1 week of free slack and can be moved 1 week to the right with a reductions of 8 in the value of ESS_A. Step 3 has reduced the value of ESS_A from 140 to 48 and the value of ESS_B frrom 134 to 58. The net result is a reduction in the value of ESS from 274 to 106. Figure 3 shows the schedule resulting from applying Step 3.

STEP 4.

Re-examination of activity 6-8 shows it can be moved 1 additional week to the right with a reductions 12 in the value of ESS_B. No other activities can be moved without increasing some ESS_j so Step 4 is terminated. Figure 4 shows the schedule at the end of Step 4. The value of ESS has been reduced to 96. Completion of this step marks the termination of the Burgess-Killebrew procedure and hence, the schedule in Figure 4 would be that also found by their algorithm.

STEP 5.

The schedule in Figure 4 has no activities that can be moved to the left and give an improvement in the value of ESS_j of any resource. Step 5-2 (movement to the right) would therefore be ignored.

ACT	RESOURCE		D	ES	S	LS	Time (weeks)																	ESS_j
	A	B					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
1～2	2	2	2	1	4	5	2A 2B	2A 2B																
1～3	4	—	2	1	0	1	4A	4A																
1～4	4	—	3	1	1	2	4A	4A	4A	FS														
3～4	1	—	2	3	0	3			1A	1A														
1～5	2	3	4	1	5	6	2A 3B	2A 3B	2A 3B	2A 3B			FS											
2～5	—	4	3	3	4	7			4B	4B	4B		FS											
4～5	2	—	5	5	0	5					2A	2A	2A	2A	2A									
3～6	2	2	4	3	8	11			2A 2B	2A 2B	2A 2B	2A 2B												
4～7	2	—	4	5	3	8					2A	2A	2A	2A		FS								
5～7	2	—	2	10	0	10										2A	2A							
6～8	—	2	3	7	8	15							2B	2B	2B					FS				
7～8	5	1	6	12	0	12												5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	
Level of resource A							12	12	9	5	6	6	4	4	2	2	2	5	5	5	5	5	5	140
Level of resource B							5	5	9	9	6	2	2	2	2	0	0	1	1	1	1	1	1	134

Figure 2

ACT	RESOURCE		D	ES	S	LS	Time (weeks)																	ESS$_j$
	A	B					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
1～2	2	2	2	1	4	5	2A 2B	2A 2B	FS															
1～3	4	—	2	1	0	1	4A	4A																
1～4	4	—	3	1	1	2	FS	4A	4A	4A														
3～4	1	—	2	3	0	3			1A	1A														
1～5	2	3	4	1	5	6	FS		2A 3B	2A 3B	2A 3B	2A 3B												
2～5	—	4	3	3	4	7			FS				4B	4B	4B									
4～5	2	—	5	5	0	5					2A	2A	2A	2A	2A									
3～6	2	2	4	3	8	11			FS	2A 2B	2A 2B	2A 2B	2A 2B	FS										
4～7	2	—	4	5	3	8					FS			2A	2A	2A	2A							
5～7	2	—	2	10	0	10										2A	2A							
6～8	—	2	3	7	8	15								FS	2B	2B	2B	FS						
7～8	5	1	6	12	0	12												5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	
Level of resource of A							6	10	7	9	6	6	4	4	4	4	4	5	5	5	5	5	5	48
Level of resource of B							2	2	3	5	5	5	6	4	6	2	2	1	1	1	1	1	1	58

ACT	RESOURCE		D	ES	S	LS	Time (weeks)																	*ESS*
	A	B					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
1~2	2	2	2	1	4	5	2A 2B	2A 2B			FS													
1~3	4	—	2	1	0	1	4A	4A																
1~4	4	—	3	1	1	2	FS	4A	4A	4A														
3~4	1	—	2	3	0	3			1A	1A														
1~5	2	3	4	1	5	6	FS		2A 3B	2A 3B	2A 3B	2A 3B		FS										
2~5	—	4	3	3	4	7					FS		4B	4B	4B									
4~5	2	—	5	5	0	5					2A	2A	2A	2A	2A									
3~6	2	2	4	3	8	11			FS	2A 2B	2A 2B	2A 2B	2A 2B	FS										
4~7	2	—	4	5	3	8						FS		2A	2A	2A	2A							
5~7	2	—	2	10	0	10										2A	2A							
6~8	—	2	3	7	8	15								FS		2B	2B	2B			FS			
7~8	5	1	6	12	0	12												5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	
Level of resource A							6	10	7	9	6	6	4	4	4	4	4	5	5	5	5	5	5	48
Level of resource B							2	2	3	5	5	5	6	4	4	2	2	3	1	1	1	1	1	46

Figure 4

ACT	RESOURCE		D	ES	S	LS	Time (weeks)																	ESS$_j$
	A	B					1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
1～2	—	2	2	1	4	5	2A 2B	2A 2B	FS															
1～3	4	—	2	1	0	1	4A	4A																
1～4	4	—	3	1	1	2	FS	4A	4A	4A														
3～4	1	—	2	3	0	3			1A	1A														
1～5	2	3	4	1	5	6	FS		2A 3B	2A 3B	2A 3B	2A 3B	FS											
2～5	—	4	3	3	4	7			FS				4B	4B	4B									
4～5	2	—	5	5	0	5					2A	2A	2A	2A	2A									
3～6	2	2	4	3	8	11			FS			2A 2B	2A 2B	2A 2B	2A 2B									
4～7	2	—	4	5	3	8					FS			2A	2A	2A	2A							
5～7	2	—	2	10	0	10										2A	2A							
6～8	—	2	3	7	8	15										2B	2B	2B	FS					
7～8	5	1	6	12	0	12												5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	5A 1B	
Resource level of A							6	10	7	7	4	6	4	6	6	4	4	5	5	5	5	5	5	36
Resource level of B							2	2	3	3	3	5	6	6	6	2	2	3	1	1	1	1	1	54

STEP 6.

This is the step in which an increase in some ESS_j is allowed if a reduction in some other ESS_j and in the total ESS results. Reapplying Step 5-1 with this modification does not result in any activity being moved. Going on to Step 5-2, we find that the earliest activity which can be moved to the right is activity 3-6. It can be moved 2 weeks to the right, decreasing the value of the ESS_A by 12 but increasing the value of ESS_B by 8. Therefore, the value of ESS is improved by a total of 4.

No further activity movement is possible either to the right or left which will improve the value of the ESS. Therefore, the solution procedure terminates. Step 6 has changed the value of ESS from 94 to 90.

The Burgess-Killebrew algorithm would have stopped with the schedule of Figure 4 and an ESS value of 96. The algorithm of this thesis has reduced the ESS value to 90.

Ⅳ. CONCLUDING COMMENTS

The algorithm presented in this thesis provides a heuristic scheduling procedure which seeks to level resource usage over the duration of the project as determined by the critical path. Slack activities are shifted within their free slack times in an attempt to reduce the total error sum of squares of all resources about their averages over their usage durations. If the user is interested in smoothing all resources on an individual basis, then the schedule obtained at the end of step 5 should be used. Step 6 provides the trade-off steps where an improvement in some resource's leveling may be gained at the loss of some other resource's leveling.

The objective function used assumed all resources were equally important. Practical situations may suggest that this is inappropriate and the k_j factors as shown in equation (4) would need to be determined. This is not a trivial problem since the error sums of squares are being compared rather than, say, peakload requirements.

Three possible improvements may be worthy of study and incorporation into this algorithm. They are;

1. moving two or more activities at the same time on the same slack path,
2. activity splitting,
3. relaxation of project duration.

During Step 6 in the solving of the example, it was found that an improvement of 28 in the value of ESS would be obtainable if activity 3-6 were moved 3 more weeks to the right from its position as shown in Figure 5. This would only be possible if activity 6-8 were moved ahead of it. The movement of both activities together cause this reduction whereas movement of activity 6-8 would have only shown an increase and would not have first been done. Further movement of other activities (activity 4-7 can be moved 1 week to the left and activity 1-2 can be moved 4 weeks to the right) would then be possible and would cause a reduction of 18 in the value of ESS and 2 weeks of usage of resource B is reduced. Therefore, the ESS would have changed from 90 down to 44. Because movement of a group of activities on the same slack path does provide additional solution complexity, it would probably be most appropriately done after the completion of Step 6 since at that point movement of any single activity would give no further improvements in the *ESS* value.

Activity splitting would allow activities to be divided into smaller lengths. Such is often possible on, for example, construction or overhaul projects; shorter "activity" durations result allowing more freedom in shifting slack activities to the right or left.

Relaxation of project duration in absence of cost penalties provides an easy way of improving on ersource leveling. Realistically, however, penalties are usually charged and thus a project cost versus project duration trade-off problem arises. The improvement in smoothing must then be priced out and compared to the penalty cost. To analyze such a problem, the smoothing algorithm above would be used to provide a smoothed schedule for several different specified durations. Then, total labor costs based on the regular time wages for the sum of the usage average $\bar{X}_j$ and some average positive error plus overtime charges above this sum for each resource exceeding it could be compared with penalty costs for each specified project duration in searching for an optimum schedule. Such an analysis contrasts with the usual PERT/COST notion of cost versus time trade-offs where activity durations can be reduced for an increase in activity cost.

APPENDIX

Burgess Leveling Procedure [5]

STEP 1.

List the project activities in order of precedence by arranging the arrow head numbers in ascending order, and when two or more activities have the same head number, list them so that the arrow tail numbers are also in ascending order. (This assumes that the network events are numbered so that activity tail numbers are always less than the head numbers.) Add to this listing the duration, early start, and slack values for each activity.

STEP 2.

Starting with the last slack activity (the one nearest the bottom of the list), schedule it to give the lowest total sum of squares of resource requirements for each time unit. If more than one schedule gives the same total sum of squares, then schedule the activity as late as possible to get as much slack as possible in all preceding activities..

STEP 3.

Holding the last activity fixed, repeat step 2 on the next to the last activity in the network, taking advantage of any slack that may have been made available to it by the rescheduling in Step 2.

STEP 4.

Continue Step 3 until the first activity in the list has been considered; this completes the first rescheduling cycle.

STEP 5.

Carry out additional rescheduling cycles by repeating Steps 2 through 4 until no further reduction in the total sum of squares of resource requirements is possible, noting that only movement of an activity to the right (schedule later) is permissible under this scheme.

STEP 6.

If this resource(s) is particularly critical, repeat Steps 1 through 5 on a different ordering of the activities which, of course, must still list the activities in order of precedence.

STEP 7.

Choose the best schedule of those obtained in Steps 5 and 6.

STEP 8.

Make final adjustments to the schedule chosen in Step 7, taking into account factors not considered in the basic scheduling procedure.

BIBLIOGRAPHY

1. Burgess, A.R. and Killebrew, J.B., "Variation in Activity Level on a Cyclic Arrow Diagram," *Journal of Industrial Engineering*, v.13, n. 2, March-April 1962.
2. Levy,F.K., Thompson, G.S., and Wiest, J.D., "Multiship-Multishop Workload-Smoothing Program," *Naval Research Logistics Quarterly*, March 1963.
3. Dewitte, L., "Manpower Leveling of PERT Networks," *Data Processing for Science/Engineering*, March-April 1964.
4. Davis, E.W. "Resource Allocation in Project Network Models-A Survey," *Journal of Industrial Engineering*, v. XVII, n. 4, April 1966.
5. Moder, J.J., and Phillips, C.R., *Project Management with CPM and PERT*, Van Nostrand Reinhold Company, New York, 1970.