A Critical Look at PERT Analysis

Richard M. Nicholson

UNIVERSITY OF MARYLAND
COMPUTER SCIENCE CENTER
COLLEGE PARK, MARYLAND

Technical Report TR-154 April 1971
NGL-21-002-008

A Critical Look at PERT Analysis

Richard M. Nicholson

ABSTRACT

PERT—Program Evaluation and Review Technique—analysis is described and the errors which can result from PERT analysis are indicated. A model for performing the PERT calculations without these errors resulting is derived and output from this model is displayed. A system for project management incorporating this model is described. A program listing of the model implemented is presented.

This research was supported in part by National Aeronautics and Space Administration Grant NGL-21-002-008, to the Computer Science Center of the University of Maryland.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 BACKGROUND

Effective management of a research and development project requires the early establishment of a project schedule and a frequent review of progress to ensure that the schedule is being met. This schedule provides the basis for the specification of a completion date and serves as a planning tool for the estimation of total cost as well as for the scheduling of material deliveries, manpower and facilities allocation, subassembly delivery dates, etc. Actual progress is compared to this schedule to detect any deviation which might occur so that a revised schedule can be produced, or action taken to correct the deviation.

Performing any such project involves the execution of lengthy sequences of individual tasks. Consequently, expected completion time is a complex function based on the expected duration of each individual task each of which is calculated with some degree of uncertainty. PERT - Program Evaluation and Review Technique - is a management tool which provides for the systematic expression of the interrelationships between the individual tasks of a project and for the statistical estimation of overall completion time given estimates of time required to perform the invidual tasks.

Although there are many kinds of jobs which involve the performance of a number of interrelated tasks, PERT is not utilized for all of these; for example, other techniques are applied to the scheduling of a job shop or assembly line activities. In the construction industry, a technique known as the Critical Path Method (CPM) is employed; the CPM approach is basically the same as that which will be described for PERT, with the exception that the time required to accomplish each task is regarded as a deterministic, rather than probabilistic, quantity. Any project composed of tasks about which significant experience and historical data are available is a strong candidate for CPM.

PERT is applied specifically in the management of those projects involving tasks which are unique to the particular project, rather than being of a routine or repetitive nature. It may be said that the planners of a project employing PERT are faced with the problem of estimating creativity; however, it is assumed that the manager or engineer responsible for providing the time estimates for a particular task has sufficient experience in similar jobs to enable him to specify a range of times in which the task may be expected to be accomplished and to provide a reasonable "most likely" time for its accomplishment.

Since individual tasks, or sequences of tasks, can frequently be performed simultaneously, there may be flexibility in the schedule time requirements for certain of the tasks.

The PERT calculations produce a schedule listing completion times for each of the tasks as well as for the entire project and indicate where this sort of flexibility exists. Given this kind of information, the manager has a relatively objective standard against which to measure progress.

PERT was developed in 1957-58 in an all-out effort to accelerate completion of the Polaris Fleet Ballistic Project, and its use has been largely credited with the successful coordination of the several thousand agencies and contractors involved, advancing the completion date by two years. PERT was subsequently applied to the Air Force's Minuteman and B-70 projects and the Army's Nike-Zeus, Pershing and Hawk projects. Continued success with PERT applications has resulted in a requirement by the Defense Department that PERT be used for research and development projects whose cost exceeds $25 million and for procurement projects over $100 million [1].

Most of the major computer manufacturers and a number of software companies have developed computer programs that accept basic individual task information, create the project schedule, and generate progress reports. These programs are for the most part little more than an automation of the hand calculated method, which is described in the second portion of this chapter.

The PERT concept is invariably implemented by using certain simplifying assumptions and by analyzing this simplified data by using the concept of "critical path". Briefly, this critical path approach assumes that there is one critical sequence of events (called the critical path) and all analysis of the project is relative to this particular sequence. Recently, researchers such as MacCrimmon and Ryavec [2] and Ringer [3] have criticized the commonly accepted techniques used in project scheduling, demonstrating that some of the simplifying assumptions of PERT tend to produce an optimistic schedule. The purpose of this thesis is to investigate the assumptions inherent in most PERT implementations, particularly that of a "critical" path, and to demonstrate a computer program which accurately reflects the interdependence of sequences of simultaneous activities in large scale projects, accounting for the effects of uncertainty in individual activities in the process of projecting overall completion times.

1.2 Standard PERT Definitions

Let us first develop some of the standard notion used in PERT.

Activities. By activity is meant an individual task which is not further subdivided, that is, activities are regarded as the basic tasks which are involved in the overall project, and become the basic building blocks of the PERT system. Activities are denoted a, b, c, ... Associated with an activity is a length of time or duration required for its completion alone. Typically this is not a fixed quantity of time, but is more accurately expressed as a statistical distribution which indicates the minimal, maximal, and expected (mean) time required for its completion.

The distribution associated with the activity a is represented as a probability distribution function (pdf), $f_a(t)$. This function is defined

$$\int_0^\infty f_a(t)dt = 1$$

and

$$pr(t \leq T) = \int_0^T f_a(t)dt.$$

The form of the pdf utilized in PERT is further defined by the minimal point m_a and maximal point M_a, so that

$$f_a(t) = 0 \text{ for } t < m_a, \quad t > M_a .$$

Clearly, the expected time of completion of a is a function of this distribution f_a and is denoted in the usual fashion $\mu(f_a)$, or more simply just μ_a.

The extent to which the component tasks of the project are broken down into subtasks, and eventually into the smallest task or activities, depends on the ability to accurately estimate these distributions f_a. Typically, any task for which f_a can be estimated with reasonable accuracy is regarded as an activity and not further analyzed.

However, it is unrealistic in practice to expect a project manager to know the form of a given activity distribution f_a. It is much more likely that he can estimate only the minimal, maximal, and usual durations, which correspond to the extremal and modal points of f_a, respectively. For this reason, all PERT implementations assume a standard form of distribution called the beta distribution and vary its parameters to yield the extremal and modal points, denoted here as m_a, M_a and $mode_a$.

The pdf of the beta distribution is

$$f_a(t) = \begin{cases} K\,(t - m_a)^{\alpha} \cdot (M_a - t)^{\beta}, & \text{for } m_a \leq t \leq M_a \\ 0 \text{ elsewhere,} & \end{cases}$$

where $\alpha, \beta > -1$.

Figure 1 illustrates a few examples of beta pdf with specified parameters.

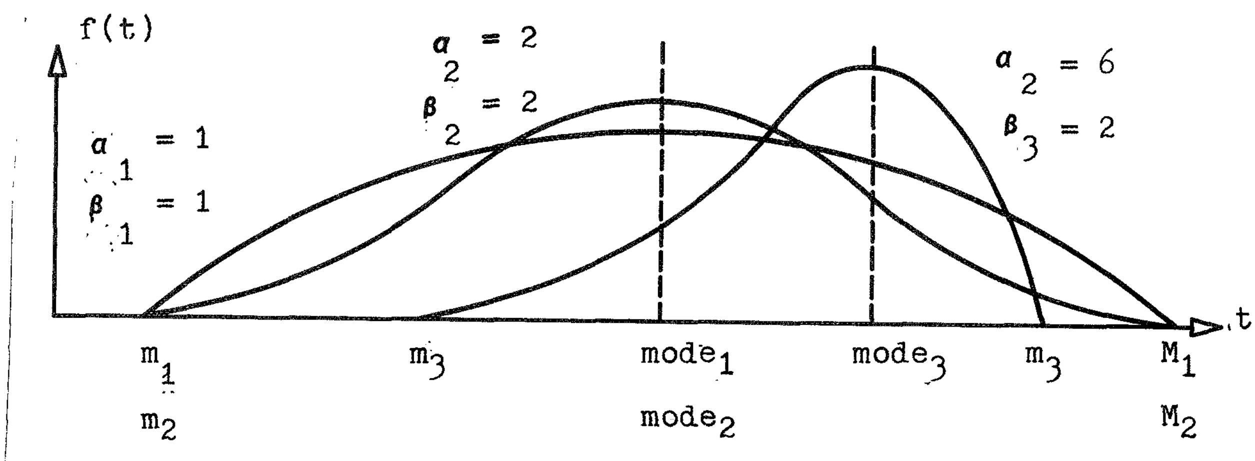


Figure 1. Examples of the Beta Distribution

The beta distribution has characteristics which seem intuitively to represent the manner in which activities may be expected to be distributed:

1) It is unimodal. There is a single most probable completion time.

2) It is continuous. "Continuity reflects the property that if an activity has a particular probability of being completed in a small interval, the probability is only slightly increased when the size of the interval is increased." 1/

3) The distribution has definite end points. The activity must consume a non-negative quantity of time, and there is assumed to be some maximum time which the activity will not exceed, barring an "Act of God."

It is condition (3) that seems to make the beta a better "standard" distribution than the better known normal distribution.

There is no attempt made to estimate the exponential parameters α and β of the beta distribution. However, as Figure 1 demonstrates, m_a, M_a and $mode_a$ are insufficient to specify a unique beta distribution; curves 1 and 2 are identical in these values and yet represent different pdf's. So the activity pdf is further defined by the assumption of a mean $\mu_a = (m_a + 4\ mode_a + M_a)/6$ and a variance $\sigma_a^2 = (M_a - m_a)^2/36$. From these two additional assumptions, a unique pdf can be assigned to activity a.

Although the existence of a beta distributed pdf is used to justify several theoretical results in the application of

---

1/ Mac Crimmon and Ryavec [2], p. 7.

PERT, in actual implementation a rather surprising simplification is made. Only information about the mean μ_a and variance σ_a^2 is retained about each activity, and only these two quantities are employed in further analysis.

One of the major aspects of this thesis will question 1) the exclusive use of beta pdf's, and 2) calculations based on the μ and σ^2 alone. We will develop a PERT implementation which is not bound by either of these restrictions, and use it to point out several anomalous situations.

<u>The network.</u> In any large project the initiation of certain tasks or activities must await the completion of other activities. For example, the assembly of power supply cannot commence until the design is completed and the components have been procured. Representation of this interdependence between activities is the second essential aspect of the PERT analysis.

We seek a convenient way to represent the time of completion aspect of activities together with their interdependence. The most rewarding structure has been a labeled directed graph or network, called the PERT network. Traditionally, a directed graph G is a point set P, together with a relation A on P denoted G = (P, A). Thus, A is a set of ordered pairs $\{(p,q) \mid p,q \in P\}$, where each ordered pair is commonly called an arc (or edge) of the graph. 2/

2/ Most graph theoretic terminology and results will be found in Busacker and Saaty [4], unless otherwise noted.

In this model the arcs {a} of G correspond to the activities of the project. The points P of the graph correspond to unique points in time which are known as <u>events</u> in PERT terminology. Events are denoted by integers.

A graph G = (P,A) is said to be labeled if there exist functions on P, or A, or both, into a set of labels. In a PERT network the arcs {a} are labeled with their associated pdf f_a (or some abbreviated representation of it such as m_a, M_a, $mode_a$, μ_a, σ_a^2) and the points i are labeled with specific times in the time continuum. 3/

The actual form of the graph, i.e., configuration of its arcs and points, reflects the interdependence of the activities. In general, an event (point) corresponds to the point in time (as yet unspecified) when an activity may be initiated or when it is completed. Events, therefore, serve as initial and terminal points of arcs in the PERT network. If an event is the terminal point of several arcs, it represents the point in time when <u>all</u> those activities have been completed. In the graphical representation, an event is identified by an integer contained within a circle, and an activity identified by a Roman letter. For example, in Figure 2 activity a is initiated by event 10 and terminated by event 20.

3/ One of the major refinements of this implementation will be to replace the scalar labels on the points by distributions.

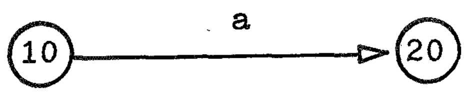


Figure 2. PERT Representation of Activity and Events

The PERT network is constructed so that all activties terminated by an event are shown entering the event, and all activities initiated by the event are shown leaving it. It has already been stated that the basic PERT assumption is that an event does not occur until all activities preceding it have been completed. It is evident that this dependence relation, together with the assumption that the project can logically be completed, implies that a partial ordering is necessarily imposed upon the events (as well as the tasks) of any PERT network. As is well known (Knuth [5] , page 259), any partially ordered set can be represented by an acyclic graph, or network, and conversely, the path relation on an acyclic graph induces a partial ordering of its points. More simply, the existence of a path from event i to event j implies that j cannot precede i, and the PERT network itself is an acyclic directed graph. Figure 3 shows a typical PERT network layout, normally called the system flow plan. This flow plan illustrates the dependency concept. Neither activity b nor activity c can be initiated until the occurrence of event 10, which is in turn dependent upon the completion of activity a.

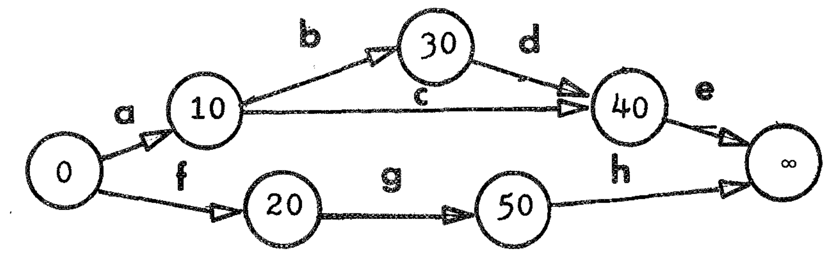


Figure 3. A System Flow Plan

From the examples in Figures 2 and 3, we notice two distinguished events identified as events 0 and ∞, which correspond to the initiation and completion of the entire project. Graphs of this configuration are called two terminal graphs, and from the problem constraints it is obvious that all PERT networks must be two terminal networks.

The typical procedure in implementing PERT is to draw the system flow plan for the project and then associate time estimates with each activity. Once these quantities have been added to the network, the entire graphic presentation is frequently referred to as a PERT chart. In the PERT chart in Figure 4, and subsequently, the three numbers $[m_a, \text{mode}_a, M_a]$ presented beneath the arc representing a will represent the extreme and modal points of the activity pdf, while (μ_a, σ_a) are placed above the arc. These five scalars become labels for the arc. This representation is standard in texts describing PERT and will be used for the present, even though it will need refinement in later sections. 4/ In all

---

4/ Note that (1) the letters associated with each arc (activity) and numbers associated with each point (event) are not labels, but merely arbitrary identifiers; (2) labels need not be scalars, and we could use the pdf's themselves (in either closed or approximate form) as labels.

calculations, the activity pdf's will be assumed independent, though clearly there are cases in which they may not be; if the duration of an activity is dependent upon its starting date (e.g., if it is dependent upon the weather), it is not truly independent of the activities preceding.

It is evident that the structure of the PERT network, together with the labels on the activities $\{ a \}$, must be given by the project description. The whole aim of PERT analysis is simply to then assign a consistent set of labels to the events. In the most simple representation these event labels are a single scalar which indicate the time at which the event occurs. In a more sophisticated representation these labels might be distributions indicating the range of possible event times.

In order to assign a consistent set of labels to the events, and in particular to event ∞ (and thus estimate the total duration of the project), it is necessary to obtain a time estimate for each of the <u>paths</u> through the PERT chart. The definition of path found in Busacker and Saaty will be employed;

> A path is a set of arcs which, if properly ordered, form a path progression. The notation $\rho(i,j)$ represents a path from event i to event j.

The concept of path length used in determining project duration differs from the normal graph theoretic definition (which is usually simply a count of the arcs involved). <u>Mean path</u>

length is the sum of the expected times of the activities which comprise the path;

$$L_{\mu}[\rho(i,j)] = \sum \mu_a \text{ for all } a \epsilon \rho \text{ , denoted } L_{\mu}(\rho).$$

Variance of path length is the sum of the variances of the activities which comprise the path;

$$L_{\sigma^2}[\rho(i,j)] = \sum \sigma_a^2 \text{ for all } a \epsilon \rho \text{ , denoted } L_{\sigma^2}(\rho).$$

Minimum path length is the sum of the minimum estimates of the activities which comprise the path;

$$L_m[\rho(i,j)] = \sum m_a \text{ for all } a \epsilon \rho \text{, denoted } L_m(\rho).$$

Maximum path length is the sum of the maximum time estimates of the activities which comprise the path;

$$L_M[\rho(i,j)] = \sum M_a \text{ for all } a \epsilon \rho \text{, denoted } L_M(\rho).$$

Because it is common to disregard the activity pdf once it has been used to calculate the mean and variance of the activity duration, only the mean path length L_{μ} and its variance L_{σ^2} are used in the typical PERT implementation. Using the mean path length over all paths from the initial event 0 to an event i in the PERT chart, the expected time for the occurrence of event i, or expected event time denoted EET_i, can be defined;

$$EET_i = \max \{L_{\mu}(\rho) \text{ for all } \rho(0,i)\}.$$

It is common in PERT analysis to call this value the early event time of event i. Some slight justification of this terminology will be given later, but in the author's opinion it is misleading. Nevertheless, we will use it to maintain a consistency with literature. In Figure 4, each activity (edge of the graph) has been labeled with its corresponding distribution f_a. (For simplicity in this and the following examples only the parameters (μ_a, σ_a^2) and $[m_a, mode_a, M_a]$ are given.) The early event time for 10, EET_{10} is eight days (all estimates will be stated in days, although clearly any unit of time may be used).

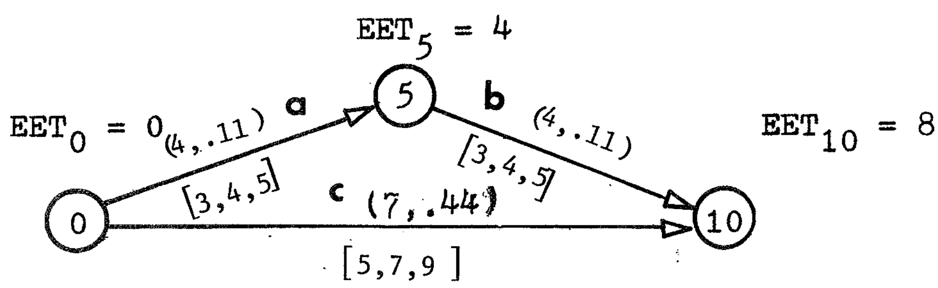


Figure 4. Calculation of Early Event Time (EET)

Although the expected time for the occurrence of event 10 is eight days after project initiation, the time estimates allow for its occurrence in six days (activities a and b requiring three days apiece, and c requiring not more than six days). On the other hand, if all activities require the maximum times estimated, event 10 will not occur until ten days have elapsed. The terms earliest possible time EPT and latest possible time LPT will be introduced to represent the extreme values. The earliest possible time for occurrence of event i is defined,

$$EPT_i = \max \{L_m(\rho) \text{ for all } \rho(0,i)\}.$$

The latest possible time for the occurrence of event i is similarly defined

$$LPT_i = \max \{L_M(\rho) \text{ for all } \rho(0,i)\}.$$

As indicated, EPT_i and LPT_i are not defined in existing PERT procedures; rather, it is assumed that EET_i is the mean of a normal distribution with a variance equal to $L_\sigma 2\ (\rho_c)$, where ρ_c is the same path that determined EET_i (any path $\rho(0,i)$ for which L_μ is less than $L_\mu(\rho_c)$ will be denoted $\rho_c{}'$).

The path $\rho_c\ (0, \infty)$ from the initial to the terminal event on which the function L_μ attains it maximum (i.e., EET_∞) is called the <u>critical path.</u> As before, the overall variance is assumed to be $L_\sigma{}^2\ (\rho_c)$. 5/ Thus, EET_i for any event in this model is the expected time for the occurrence of event i, and in particular EET_∞ is the expected time for the completion of the entire project. Thus it is reasonable for the project manager to set EET_∞ as an overall project goal. With this in mind, we may ask how late any event can occur and still meet the overall goal of EET_∞. So <u>late event time</u> LET_i is defined as the difference between the overall project completion date EET_∞ and the expected length of time required to complete all activities subsequent to event i;

$$LET_i = EET_\infty - \max \{ L_\mu(\rho) \text{ for all } \rho(i,\infty) \}.$$

<u>Slack.</u> If $EET_i = LET_i$, then

$$\max \{L_\mu \text{ for all } \rho(0,i) \} = EET_\infty - \max \{L_\mu \text{ for all } \rho(i,\infty)\}$$

so that

$$\max \{(L_\mu \text{ for all } \rho(0,i)\} + \max \{L_\mu \text{ for all } \rho\ (i, \infty)\} = EET_\infty$$

5/ If more than one path is found with L_μ equal to its maximum, the path with the largest variance $L_\sigma{}^2$ is taken to be the critical path.

$= \max \{ L_\mu \text{ for all } \rho(0, \infty) \}$,

and event i is found to be on the critical path. Otherwise $LET_i > EET_i$, meaning there is a grace period between the expected time and the deadline (i.e., time which still assures reasonable probability of meeting the overal project goal EET_∞) for the event. This period is called slack, and all events not on the critical path have positive slack. It is now evident what is "critical" about the critical path, and why PERT analysts pay particular attention to the critical path. The slack of an event represents a buffer which will absorb some slippage in the completion of activities on paths passing through the event, and there is no slack on the critical path; consequently, a delay in the completion of an activity on the critical path will be expected to result in project overrun.

The term early event time for EET_i appears to have been chosen to contrast with late event time LET_i in this determination of slack. Further, it is apparent that in this implementation of PERT two consistent labels for each event (point) of the network are determined and that their difference is the slack.

The late event times of the PERT network present themselves as appropriate project milestones. If the late event time LET_i for any event i is not realized, there is a high likelihood of a delay in project completion. It is also apparent that one of the paths $\rho(i, \infty)$ now becomes a "critical" path. There may now be several paths on which the events contain negative slack ($LET_i < EET_i$); if the original schedule EET_∞ is to be met, all such paths must be shortened by the acceleration of one or more activities of each. The PERT chart will indicate which of the paths, if any, still contain slack, so that the project manager may determine whether any resources are available for shifting from activities on these paths to activities on the now critical path(s).

1.3 Analysis by the Critical Path Method

Since the entire goal of the critical path analysis in PERT is to assign EET_i and LET_i in each event, let us illustrate the procedure by an example.

Figure 5 shows a subnetwork of a PERT chart of a major research and development project. Labels beneath the arcs are consistent with the notation introduced earlier; for each activity, extremal and modal estimates of completion time are listed. Using the expressions for mean $\mu_a = (m_a + 4\ \text{mode}_a + M_a)/6$ and variance $\sigma_a^2 = (M_a - m_a)^2/36$, these values are calculated for each activity in Figure 6.

The next step in a critical path analysis is the determination of the early event time EET_i for each event; this operation is called the forward iteration or <u>forward pass.</u> EET_i will be show above event i enclosed in a square. Thus, the early event time for event 10 is shown as $\boxed{6}$ in Figure 7. This quantity is found by adding the 6 days required to

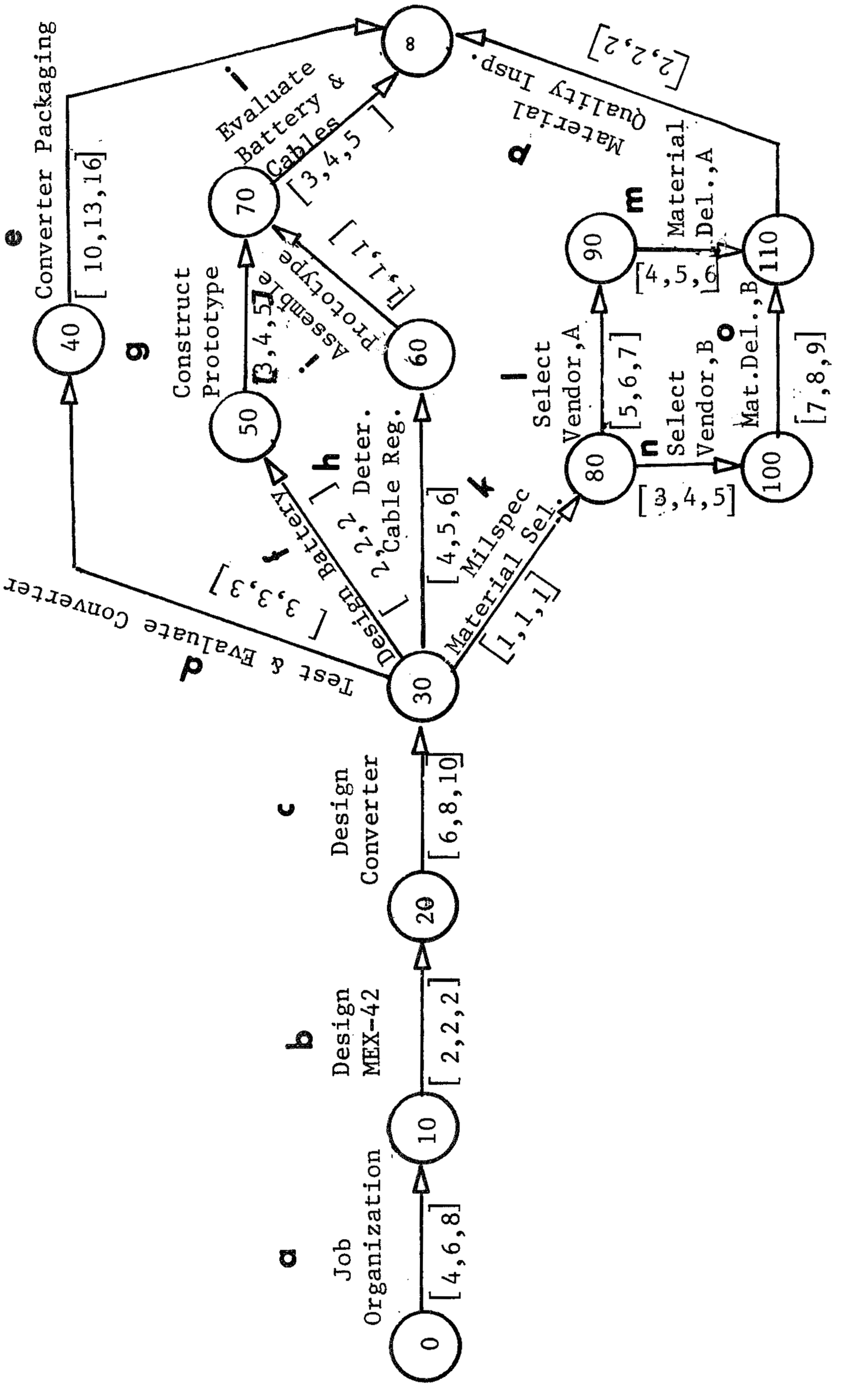


Figure 5. Network for Calculation of Early Event Times and Late Event Times

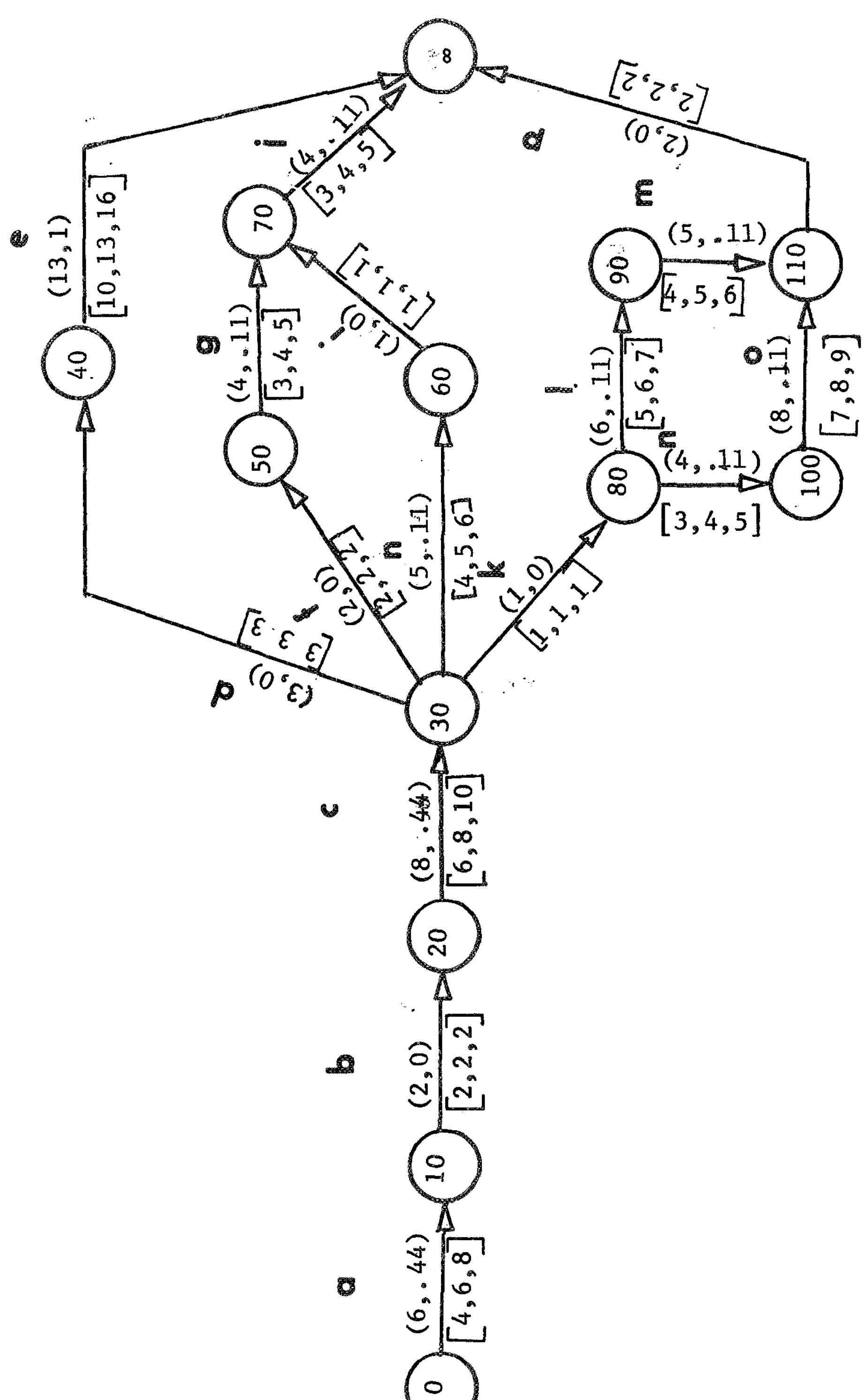


Figure 6. Calculation of Activity Expected Times and Variances

complete activity a to the EET_0 which is always taken to be 0. Activity b is expected to consume 2 days beginning 6 days after project initiation, so its completion (event 20) should occur at 6 + 2 = 8 days. Similar calculations may be used to determine the early event times for events 30, 40, 50, 60, 80, 90 and 100 because each has only one path leading to it from event 0. These times are shown in Figure 7.

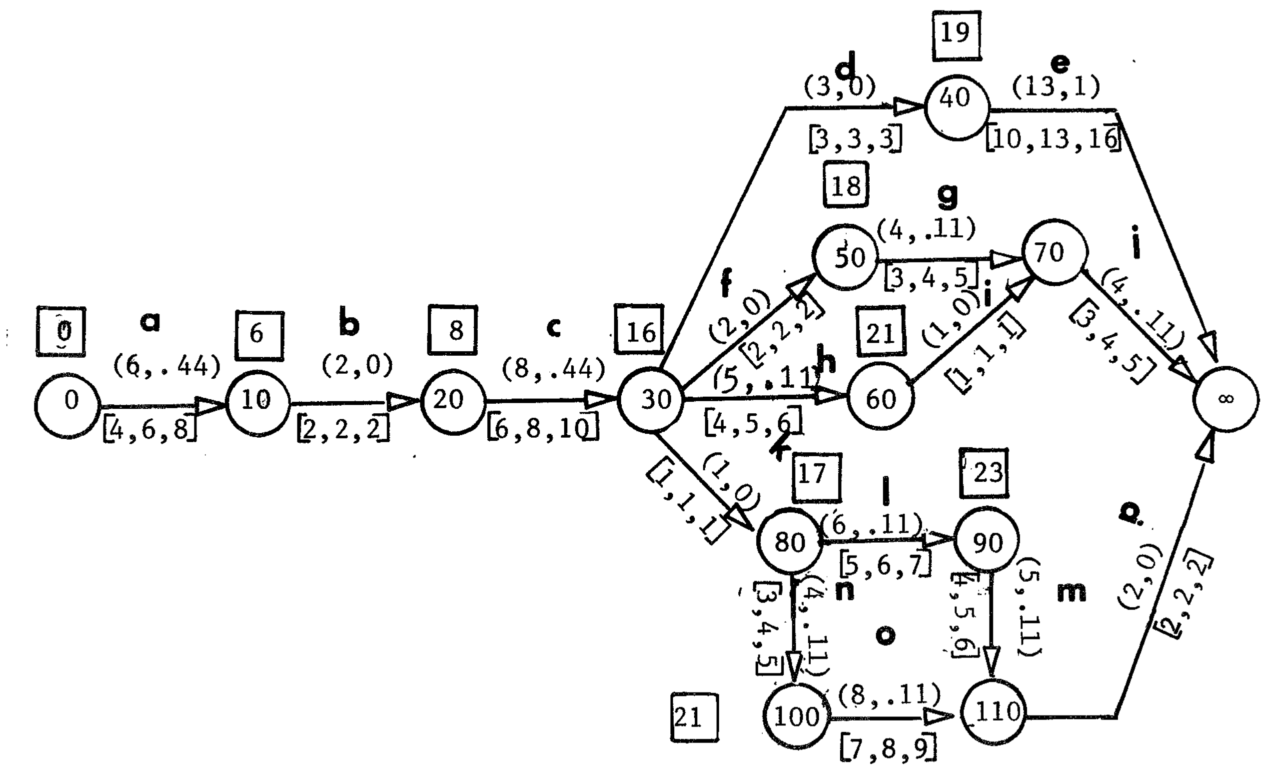


Figure 7. Early Event Times with
No Parallel Paths Considered

Given EET_{90} and EET_{100}, we can calculate EET_{110}. ρ_c (0,110) is the path passing through event 100 - its length is 29 days, as compared to a length of 28 days for the path passing through event 90, so EET_{110} is 29 days. Continuing through the network, EET_i is calculated for each event i, after early event times on all the paths $\rho(0,i)$ have been determined. Figure 8 shows early event times for each event in the network. EET_∞ is 32 days, with the critical path consisting of activities a, b, c, d, and e.

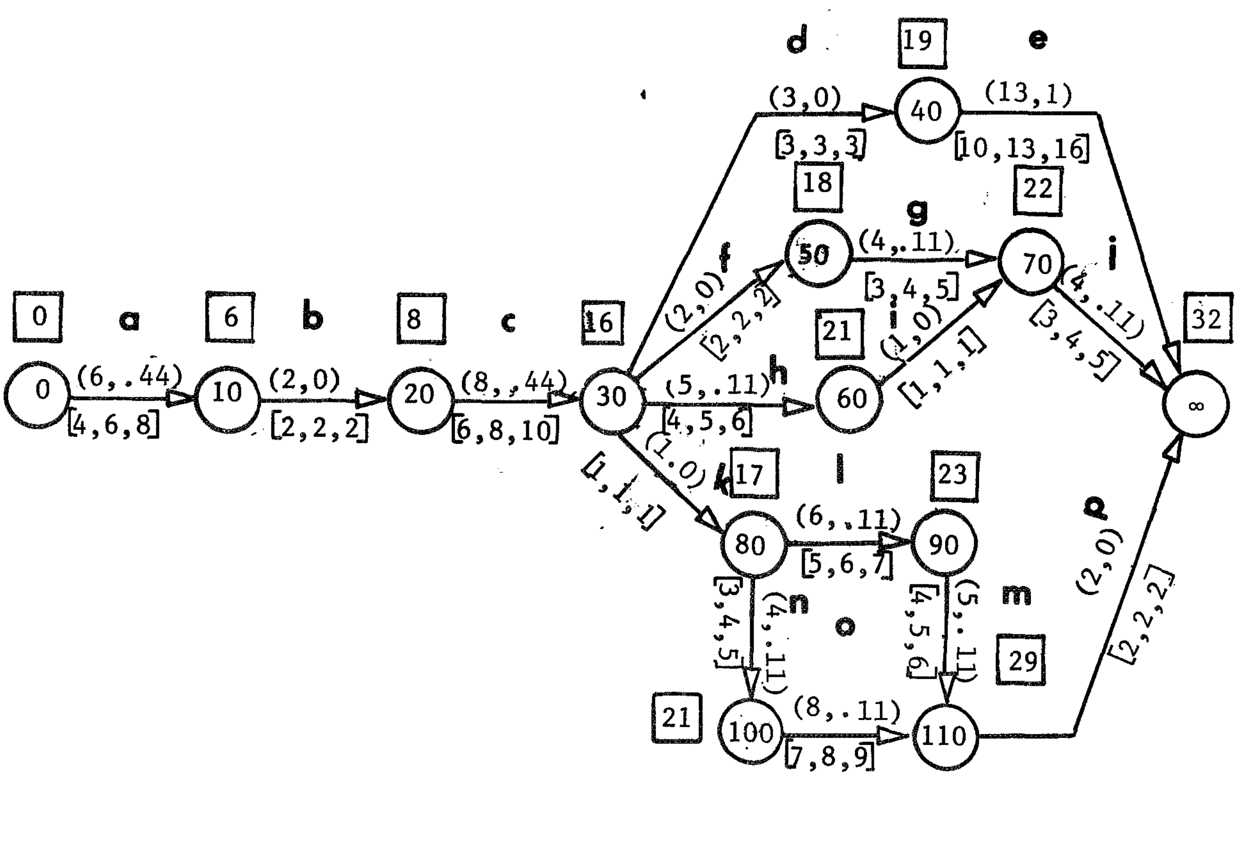


Figure 8. PERT Network with All Early Event Times Calculated

If the overall project target date is set at 32 days, LET_∞ is 32. The expected time to perform activity j is 4 days, so $LET_j = 32 - 4 = 28$. The late event time for an event is shown in Figure 9, enclosed in the circle above the early event time. Calculation of the late event times in an operation known as the backward pass is analogous to the forward pass, with LET_∞ serving as the initial time for calculations as EET_0 did for the forward pass. Thus, the calculation of LET_{80} is dependent upon the path through event 100; late event time LET_i for any event i can be found only after calculation of late event times for all events on all paths $\rho(i, \infty)$. Figure 9 shows the early and late event times for all events in the PERT chart; slack may be determined for any event by subtracting the early event time from the late event time.

The path $\rho(0, 10, 20, 30, 40, \infty)$ was found to be critical path with a length $L_\mu(\rho)$ of 32 days. As will be shown in Chapter III, this quantity accurately reflects the expected time to complete all of the activities on this path in sequence, assuming accurate time estimates for the individual activities. Thus, if the project were to be executed a number of times or "trials", with values of the individual activities drawn randomly from their pdf's, the mean value that would be obtained for the length of this path is 32 days, with the values obtained on various trials ranging over the interval of 25 to 39 days.

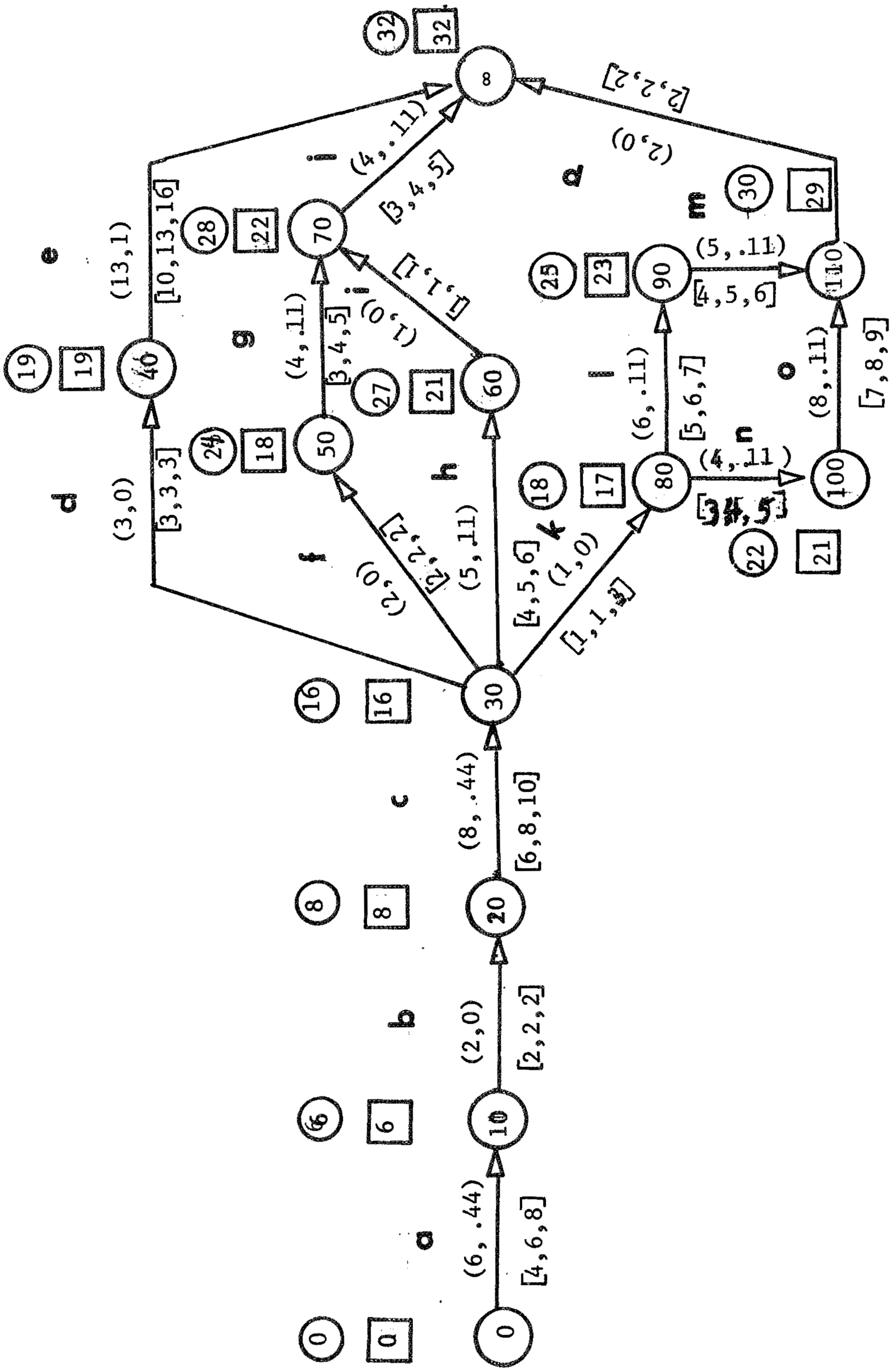


Figure 9. PERT Network Showing Early and Late Event Times

However, if we examine the path consisting of activities a, b, c, k, n, o, and p, whose expected value is 31 days but whose extreme values are 25 and 37, we see that there is considerable overlap in its range of values with that of the critical path. In the performance of the series of trials of the project, it is reasonable to expect that for certain trials a larger value would be obtained on this path than on the "critical" path. If, then, for each trial the larger of the two path values is chosen to represent the completion of the entire project for this trial, we are in effect simulating the performance of a project containing these two paths over a number of trials. Because for each trial we have assigned to project completion time the larger of the values representing completion of the two paths, the mean value for project completion may be expected to be greater than the mean value found for the "critical" path. As more paths nearly equal in length to the "critical" path are added to the project, the probability that the overall project completion time exceeds the length of any of the paths continue to increase. Thus it appears that if the individual path lengths are treated as distributions (which is consistent with the assumption that activity times are distributions) rather than deterministic values, the project duration calculated may in some cases be different from that obtained in ordinary PERT analysis. Ringer [3] has proposed that Monte Carlo simulation techniques be utilized to account for the effects of individual activity distributions; however, we intend to present a model for accomplishing this without resorting to repeated trials.

We have seen here how the typical PERT analytic techniques may be applied to project scheduling, but we have also seen that the completion date predicted by the use of these techniques might fall earlier than the time predicted by other statistical methods. In Chapter II, an example wherein the PERT-calculated completion time is clearly less than that calculated by analytical means will be presented, and additional questions will be raised concerning the individual activities.

CHAPTER II

ANOMALIES ARISING FROM CRITICAL PATH ASSUMPTIONS

2.1 Effects of Parallel Path Distributions

As we noted in the description of the PERT network, an event i is considered to occur only when all activities on <u>all</u> paths $\rho(0,i)$ have been completed. That is, the time of occurrence of i is the <u>maximum</u> time $L(\rho)$ obtained on all paths $\rho(0,i)$. The expected time of occurrence of i is then

$$E(i) = E[\max\{L(\rho) \text{ for all } \rho(0,i)\}].$$

If $\rho_c(0,i)$ is clearly longer than any $\rho_c'(0,i)$, that is, if $L_m(\rho_c) > \max\{L_M(\rho_c')\}$, then $E[\max(L_\rho)] = E\{L(\rho_c)\} = L_\mu(\rho_c)$. In this case, it is not necessary to consider the pdf's of the individual paths $\rho(0,i)$. The simplification made in PERT analysis is that the paths $\rho_c'(0, i)$ <u>never</u> contribute to the determination of μ_i. However, as the example in the previous chapter suggests, the addition of other paths whose pdf's overlap that of ρ_c can cause the expected time of event i to fall later than the time calculated using only $L_\mu(\rho_c)$. The following example demonstrates quantitatively the effect on μ_i caused by the inclusion of a second path $\rho(0,i)$ whose pdf is identical to that of $\rho_c(0,i)$.

The network [6] shown in Figure 10 contains two critical paths $\rho_c(0,30)$; the mean path lengths $L_\mu(\rho_c)$ and variances

---

6/ This example is from MacCrimmon and Ryavec [2].

$L_{\sigma}^{2}(\rho_c)$ are identical. While each path is shown to be composed of only two activities, any of the activities a, b, c, d may be thought of as paths whose durations are represented by the pdf's shown. In addition, there may be other paths $\rho_c'(0,30)$, but it will be assumed that $\max\{L_M(\rho_{c'})\} < L_m(\rho_c)$. To simplify the calculations, discrete probabilities are used to represent the activity pdf's; these are shown in Figure 10.

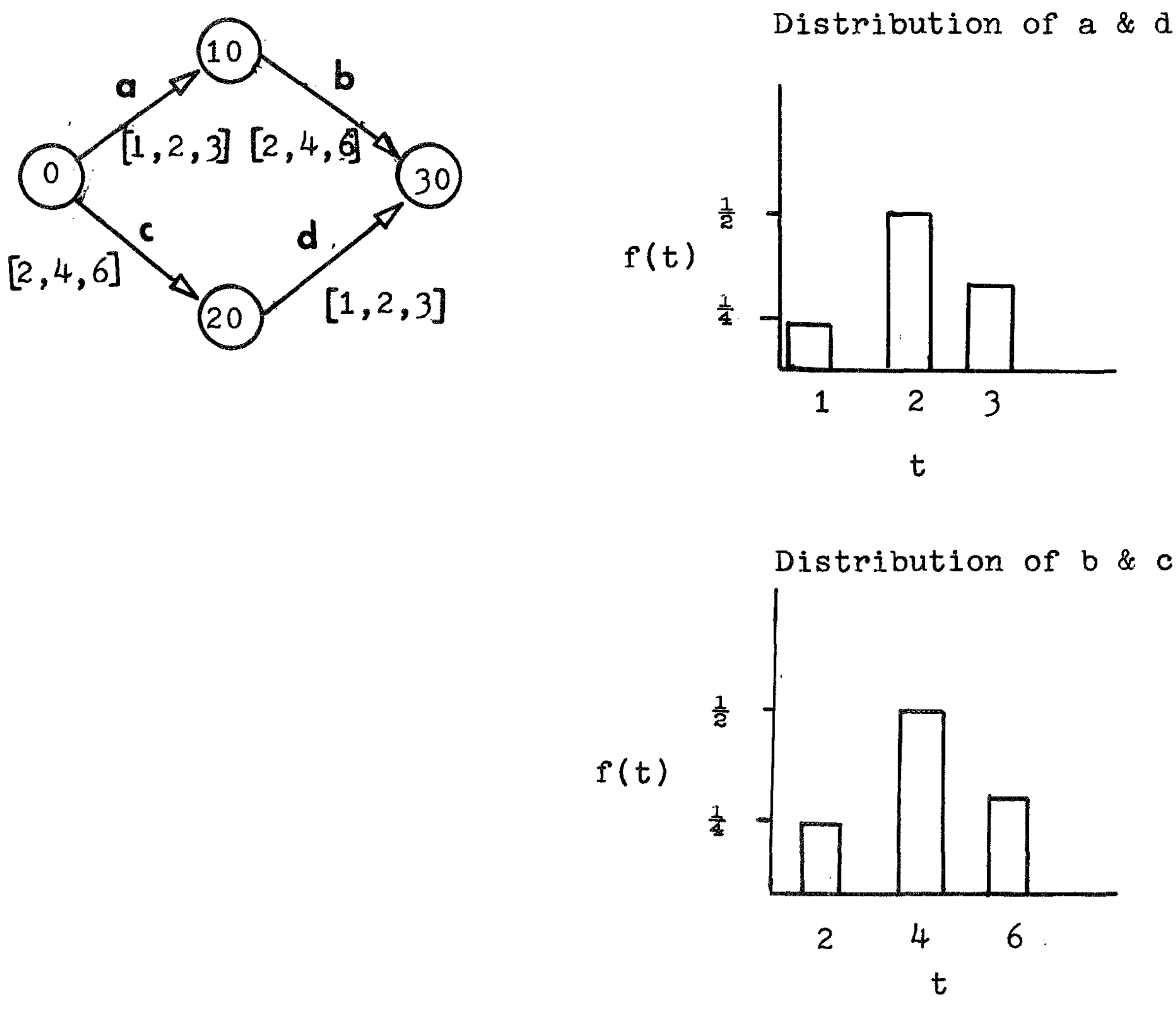


Figure 10. PERT Network with Discrete Activity Distributions

The pdf of the path $\rho(0, 10, 30)$ is determined as follows. The pdf of event 10 is that of activity a, so the path pdf is a composition of f_a and f_b in series. Table 1 is used to perform this calculation. In each row of i of table 1a, the value a_i is given for activity a and b_i for activity b. The sum of a_i and b_i is listed in column 5; $f(a_i \text{ and } b_i) = f(a_i) \cdot f(b_i)$ is listed in column 6. The pdf of activity a in series with activity b is then found using

$$f_{a.b}(t) = \sum_{a_i + b_i = t} f(a_i b_i),$$

and listed in Table 1b. The mean of this path is found by

$$\mu_{a.b} = \sum_j t_j f_{a.b}(t_j)$$

and is equal to 6 days, the value found using standard PERT methods. Because this path is "critical", $L_\mu(\rho_c) = 6$.

Calculation of the f_{30} is demonstrated in Table 2. The pdf's of the two paths $\rho_c(0,30)$ are listed across the top and down the sides of Table 2a. In the table, position (i,j) is the maximum of t_i and t_j. The probabilities of the individual combinations of t_i and t_j (not shown) are found by taking the products of the individual probabilities $f(t_i)$ and $f(t_j)$. As before, the probability for each t_i in the pdf is found by summing the probabilities of all entries with value t_i in Table 2a. Probabilities of the individual values for event 30 are shown in Table 2b, and the expected time to complete event 30 is found to be about 14.8% greater than $L_\mu(\rho_c)$.

a. Sum

a_i	$f(a_i)$	b_i	$f(b_i)$	Sum a_ib_i	$f(a_ib_i)$
1	1/4	2	1/4	3	1/16
1	1/4	4	1/2	5	2/16
1	1/4	6	1/4	7	1/16
2	1/2	2	1/4	4	2/16
2	1/2	4	1/2	6	4/16
2	1/2	6	1/4	8	2/16
3	1/4	2	1/4	5	1/16
3	1/4	4	1/2	7	2/16
3	1/4	6	1/4	9	1/16

b. Distribution of sum.

t	3	4	5	6	7	8	9
f(t)	1/16	2/16	3/16	4/16	3/16	2/16	1/16

mean = 6.0

Table 1. Distribution of Activities in Series.

a. Max

f(t)		1/16	2/16	3/16	4/16	3/16	2/16	1/16
	t	3	4	5	6	7	8	9
1/16	3	3	4	5	6	7	8	9
2/16	4	4	4	5	6	7	8	9
3/16	5	5	5	5	6	7	8	9
4/16	6	6	6	6	6	7	8	9
3/16	7	7	7	7	7	7	8	9
2/16	8	8	8	8	8	8	8	9
1/16	9	9	9	9	9	9	9	9

b. Distribution of maximum.

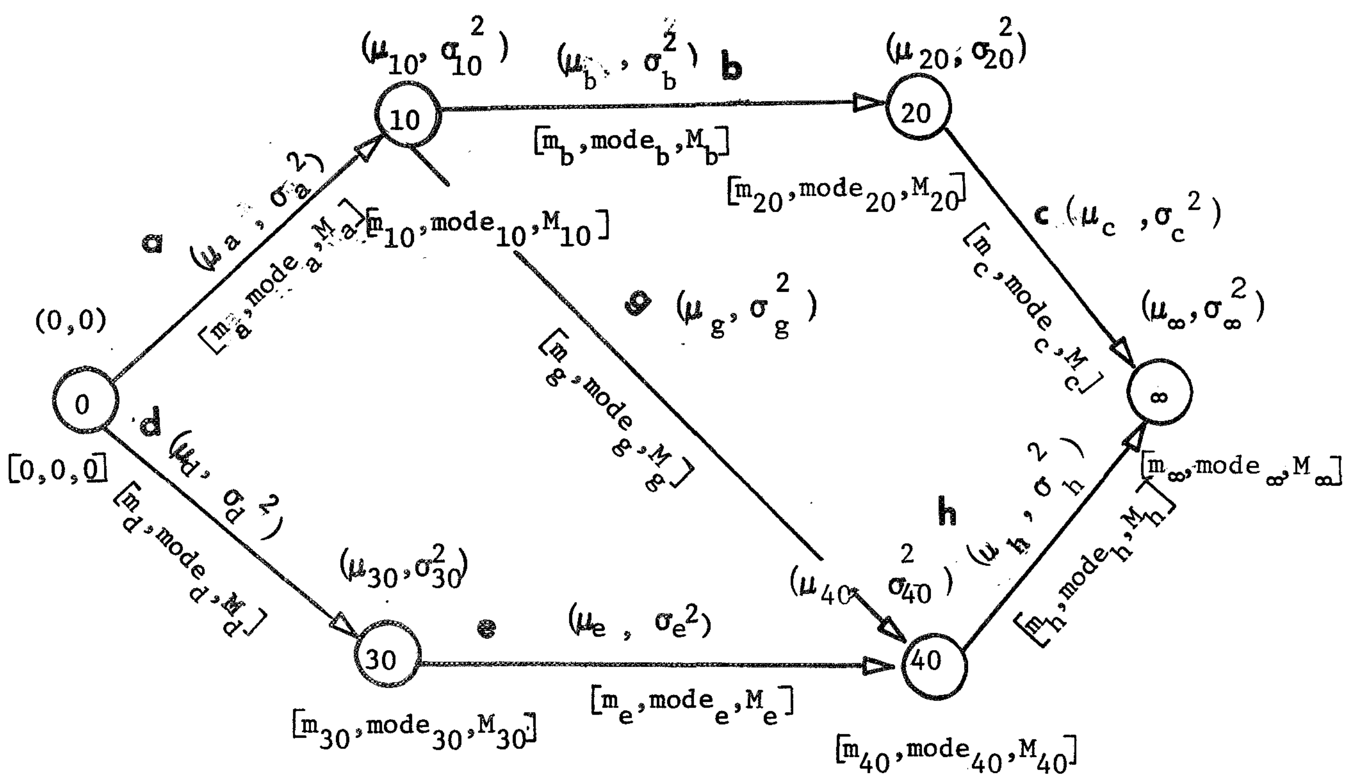
t	3	4	5	6	7	8	9
f(t)	1/256	8/256	27/256	64/256	69/256	56/256	31/256

mean = 6.89

Table 2. Distribution of Paths in Parallel.

This rather simple example, showing that the expected time of an event i (EET_i) need not be a simple function on $L\mu(\rho)$ for all paths $\rho(0,i)$, suggests an alternate implementation of the PERT concept. In fact, we know that there is really a probability distribution function associated with each event of a PERT network. The EET_i for any event, however it is calculated, serves at best as a crude approximation of this pdf. It would be more meaningful to store the pdf itself at this point.

A reasonable approach to the labeling of events is to follow the procedure used for the activities. We could assume a standard form for the pdf and define each by providing the [min, mode, max] and (μ, σ^2). So a PERT chart might have the following appearance:



Certain of these labeling parameters are easily calculated, e.g.,

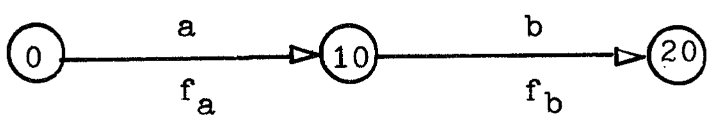
$$m_i = EPT_i$$
$$M_i = LPT_i$$

But calculation of the really essential parameters remains elusive even if all of the activity pdf's are given in closed form as in the following example.

A simple network can be constructed given two activities a, b, both with uniform pdf's

$$f_a(t) = \begin{cases} 0 & , \ t < m_a \\ \dfrac{1}{M_a - m_a} & , \ m_a \le t \le M_a \\ 0 & , \ t > M_a. \end{cases}$$

If the activities are linked in series as in the following diagram,



then the probability of event 10 occurring at time t is precisely $f_a(t)$. The probability of event 20 occurring at time t is evidently

$$pr_{20}(t) = pr(t_a) \text{ and } pr(t_b) \text{ for all } t_a, t_b \text{ such that}$$
$$t_a + t_b = t, \text{ or}$$
$$f_{a \cdot\cdot b}(t) = \int\limits_{m_a \le t_a \le t - m_b} f_a(t_a)\, f_b(t - t_a)\, dt_a.$$

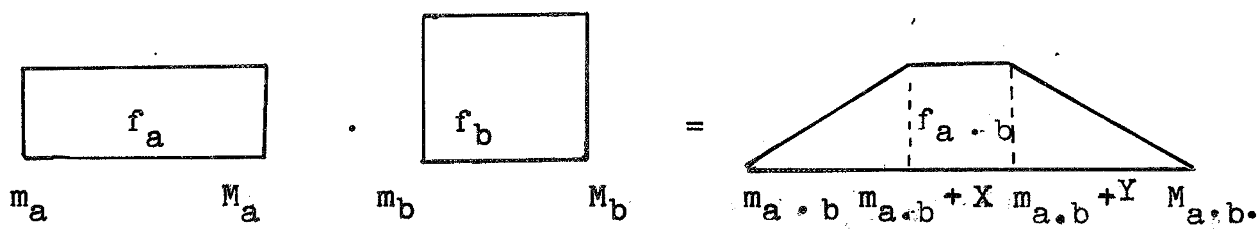
And for this example of f_a and f_b rectangular distributions,

$$f_{a \cdot b}(t) = \int_{m_a}^{t - m_b} \frac{1}{M_a - m_a} \quad \frac{1}{M_b - m_b}\, dt_a$$

If we let $X = \max(M_a - m_a, M_b - m_b)$, $X = \min(M_a - m_a, M_b - m_b)$, $m_{a \cdot b} = m_a + m_b$, $M_{a \cdot b} = M_a + M_b$, then

$$f_{a \cdot b}(t) = \begin{cases} 0 & , t < m_{a \cdot b} \\ (t - m_{a \cdot b})/XY & , m_{a \cdot b} \leq t < m_{a \cdot b} + X \\ 1/Y & , m_{a \cdot b} + X \leq t < m_{a \cdot b} + Y \\ (M_{a \cdot b} - t)/XY & , m_{a \cdot b} + Y \leq t \leq M_{a \cdot b} \\ 0 & , t > M_{a \cdot b} \end{cases}$$

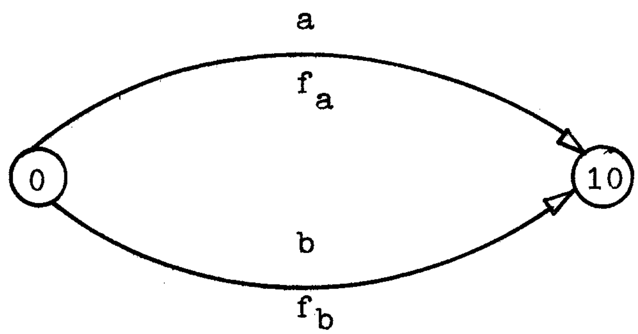
Or pictorially,



In accordance with standard PERT assumptions $E(f_{a \cdot b}) = E(f_a) + E(f_b)$ and $\sigma^2(f_{a \cdot b}) = \sigma^2(f_a) + \sigma^2(f_b)$, so that the calculation of these quantities by summing over the specified paths is evidently consistent with the PERT approach.

If, however, the two activities are related in parallel [7] as in the following network:

7/ A PERT network is seldom constructed with a single pair of events joined by two activities, but either a or b may be regarded as a path composed of more than one activity with no effect on the results presented here.



then the probability that event 10 will occur precisely at time t is evidently

$$pr_{10}(t) = pr(t_a = t) \text{ and } pr(t_b \le t) + pr(t_a \le t) \text{ and } pr(t_b = t), \text{ or}$$

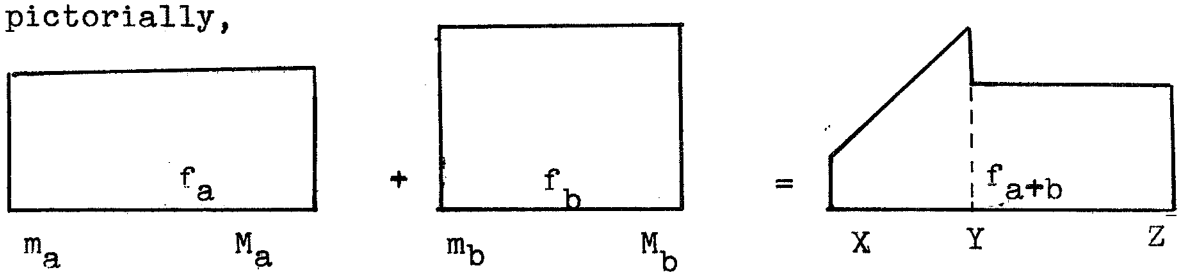
$$f_{10}(t) = f_a(t) \int_{m_b}^{t} f_b(t_b)dt_b + f_b(t) \int_{m_a}^{t} f_a(t_a)dt_a.$$

Again, when f_a and f_b are uniform distributions, we may let

$$X = \max(m_a, m_b),\ Y = \min(M_a, M_b),\ Z = \max(M_a, M_b),$$

$$f_{a+b}(t) = \begin{cases} 0 & , t < X \\ \dfrac{2t-(m_a + m_b)}{(M_a - m_a)(M_b - m_b)} & , X \le t < Y \\ \dfrac{1}{Z - Y} - \dfrac{Y^2 - Y(m_a + m_b) + m_a m_b}{(Z - Y)(M_a - m_a)(M_b - m_b)} & , Y \le t < Z \\ 0 & , t > Z \end{cases}$$

or pictorially,

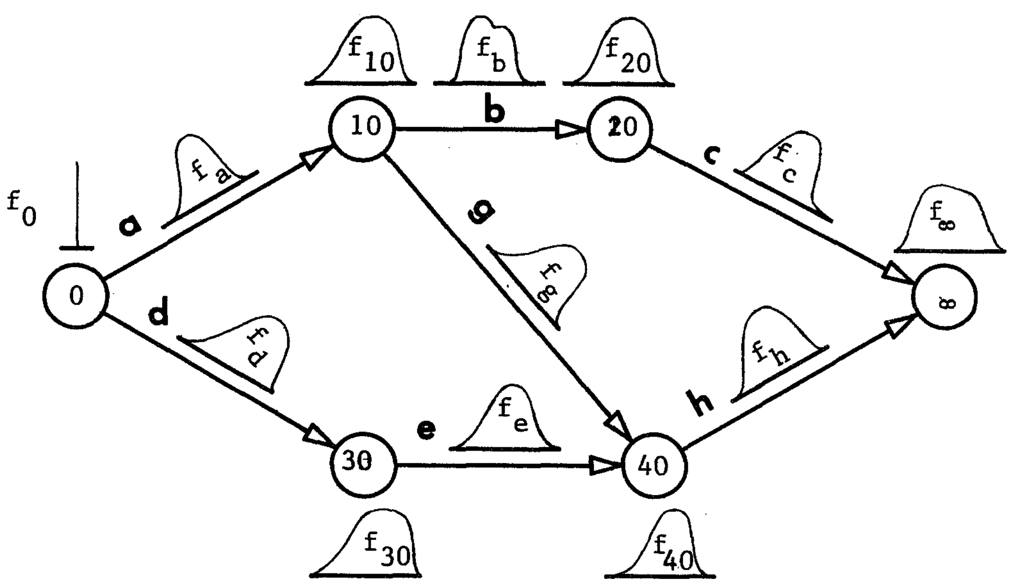


(assuming $m_a < M_b$, $m_b < M_a$. If $m_b \geq M_a$, then evidently $f_{a+b} = f_b$; if $m_a \geq M_b$, $f_{a+b} = f_a$).

These results help to illustrate, perhaps, why the designers of PERT despaired of any attempt to derive the event pdf's as a function individual activity pdf's and chose instead to define the mean of $f_{a \circ b}$ as the sum of μ_a and μ_b, and the mean of f_{a+b} as the maximum of μ_a and μ_b. Even in this extremely simple case of uniform activity pdf's the resulting event pdf's are hopelessly unwieldy. And any calculations based on these derived distributions could be expected to yield even more cumbersome pdf's.

A further demonstration of the different event pdf's that may be derived in solution of the network are presented in Figure 11 through 14, which are plots of the pdf's resulting when two input pdf's (identified as Distribution A and Distribution B) are composed in parallel or in series. The generalized computer routines which were used to perform the parallel and series compositions are those used in the implementation described in Chapter IV; they are derived in Chapter III. In Figure 11, two uniform distributions are combined in parallel, as in the above example. Figure 12 demonstrates the composition of two roughly triangular distributions in parallel; the resulting pdf is shifted noticeably to the right, indicating an increase in expected time. The next two plots, Figures 13 and 14, illustrate a situation in which the output distribution contains more modes than the input. In Figure 13, the series composition of two bimodal distribtions produces a trimodal resultant, and in Figure 14, a bimodal distribution results when two unimodal distributions are composed in parallel.

These plots demonstrate that even if all the activity pdf's are of some standard form (e.g., normal or beta distribution), there is no assurance that the resultant event pdf's need be of the same form, or even that they be unimodal, and there is no simple analytic technique for obtaining them. If, however, the algorithms necessary to numerically derive resultant distributions from some arbitrary input distributions are developed, the distributions themselves may be stored as labels in the PERT network. The network would then have the appearance,



This figure deviates from that shown in th text because of interpolation in the plotting routines

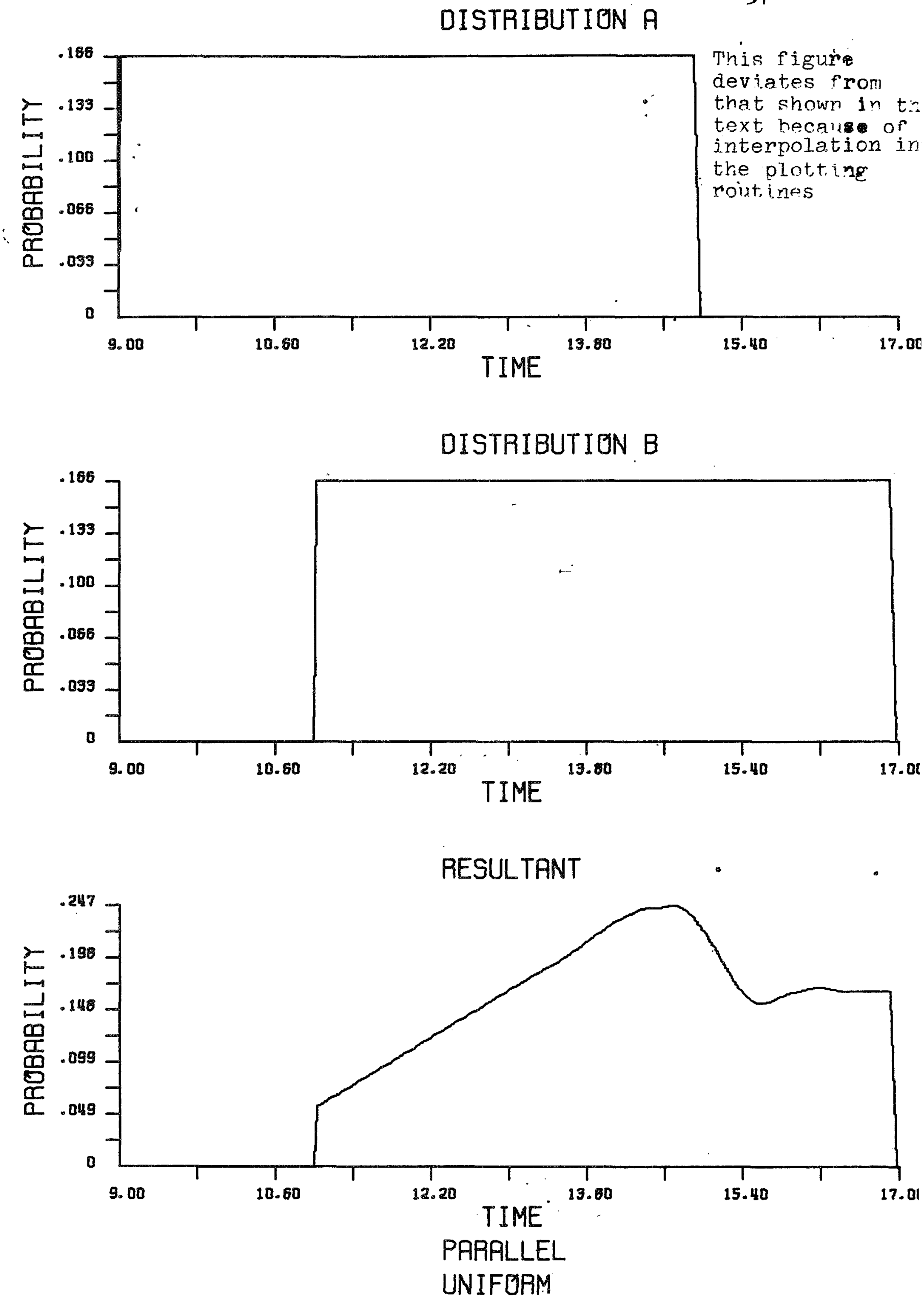
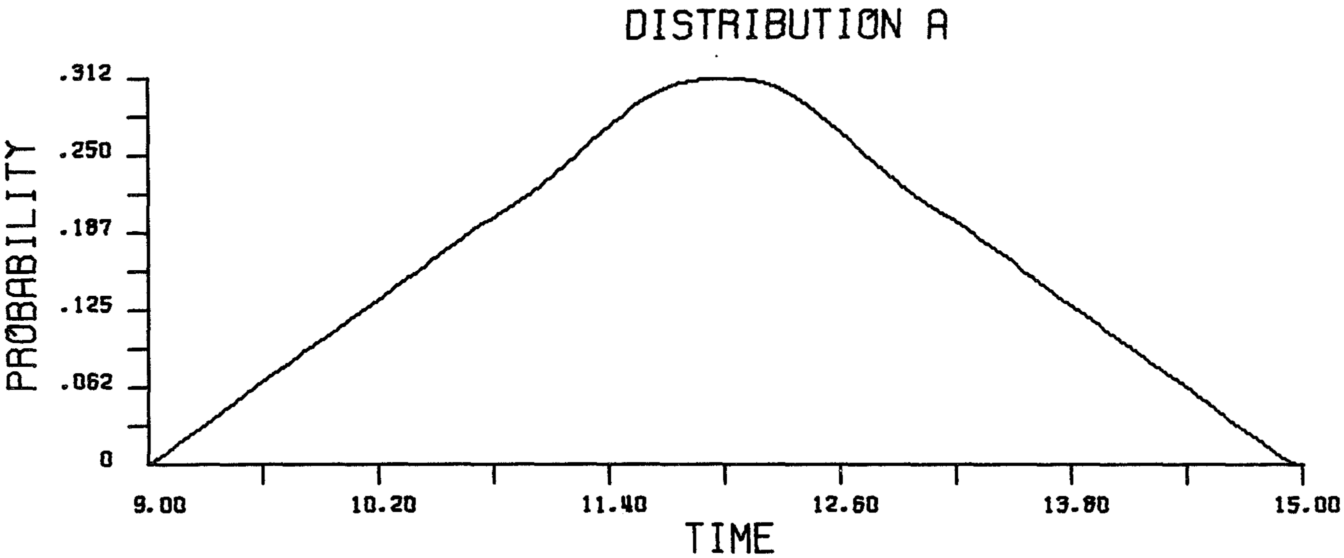
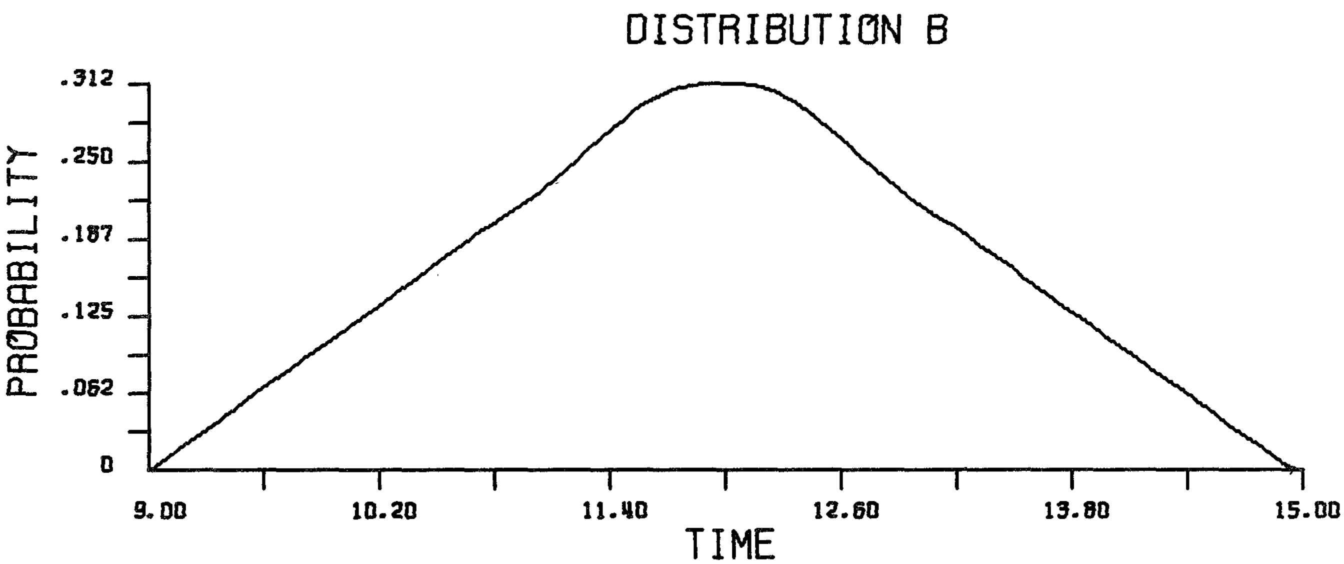
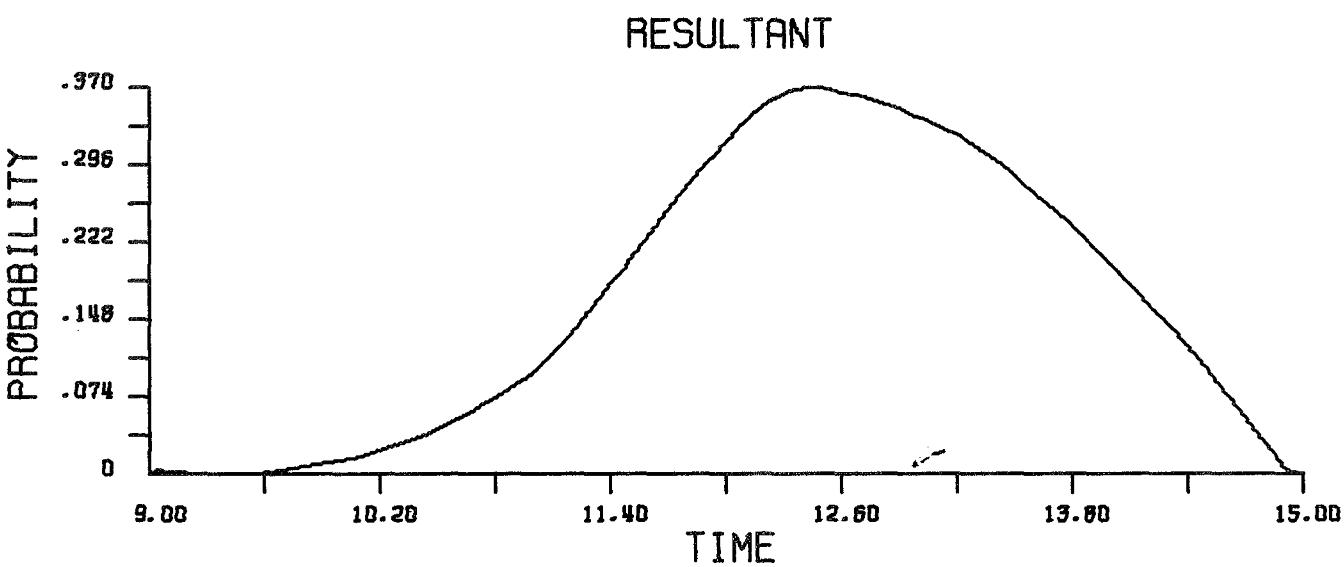


Figure 11. Parallel Combination of Two Uniformly-Distributed Activities







PARALLEL
TRIANGLE

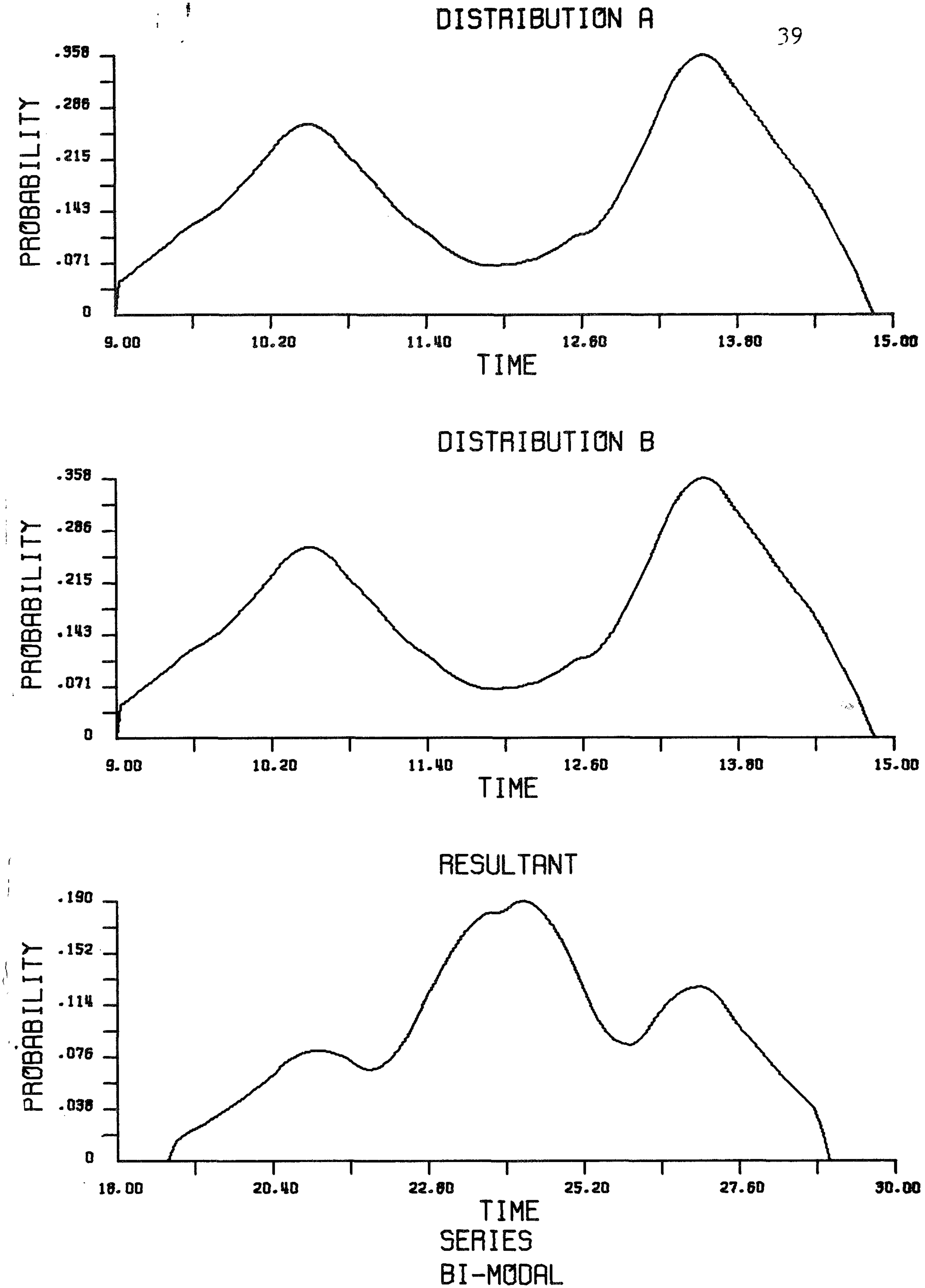


Figure 13. Series Combination of Two Activities Having Bimodal Distributions

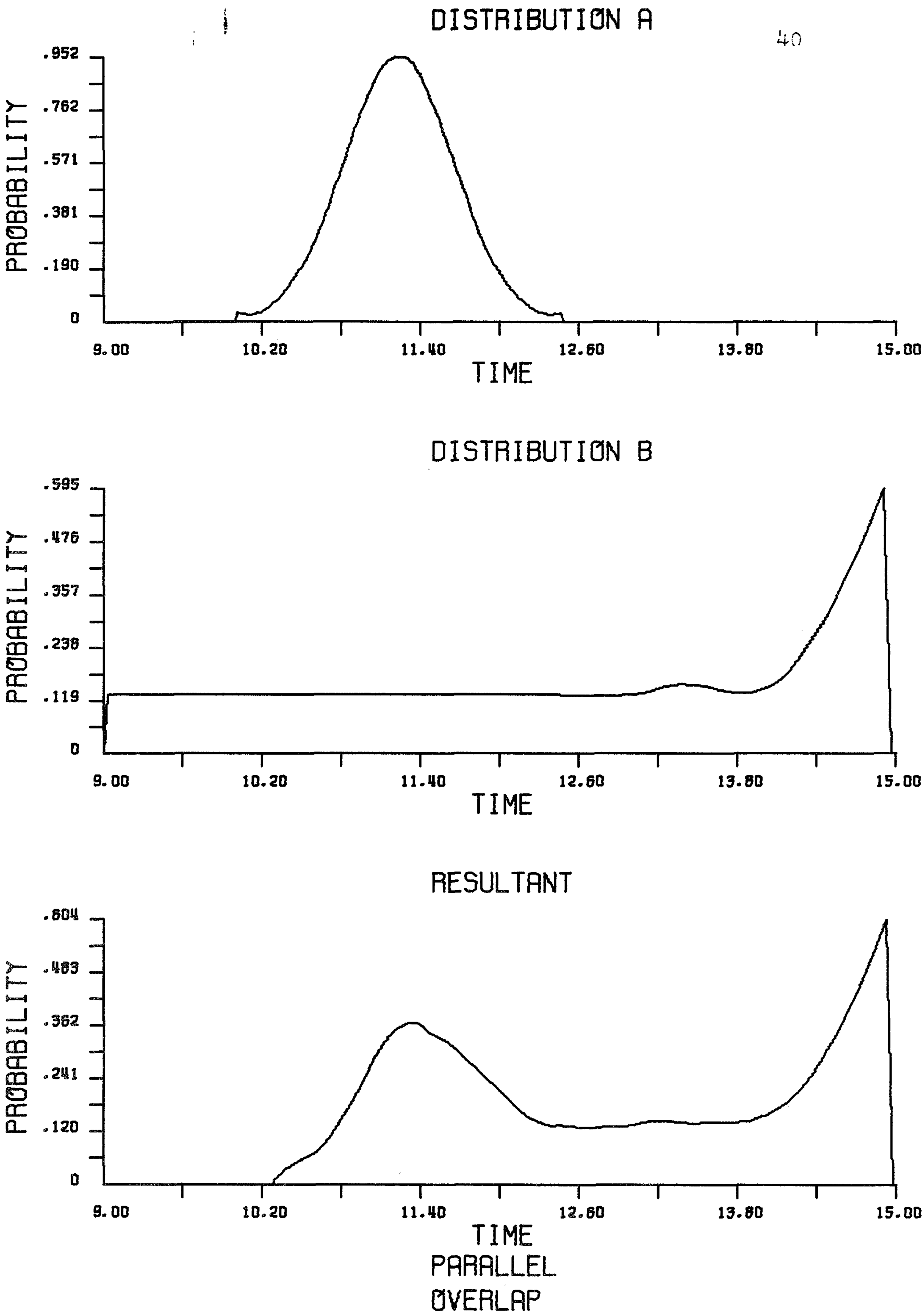


Figure 14. Bimodal Resultant of Two Activities Combined in Parallel

In Chapter IV a computer model which is a direct realization of this idea is introduced. Actual distributions will be associated with both activities and events as labels.

2.2 Activity Distributions

The examples chosen here to demonstrate the combinatorial anomalies of PERT include activities whose pdf's are other than the PERT-assumed beta distribution. While the beta distribution is a convenient form for representing the estimated time to perform many activities, there may be instances in which the best estimate of activity duration is better represented by some other distribution form.

An example of a bimodal distribution might arise in the case of an activity which requires the utilization of equipment which could be made available on either of two dates; if the material is not ready for use of the equipment by the earlier date, it is possible that progress on the activity might be suspended until the later date. In this case we would expect the activity pdf to be concentrated around two times based on these availability dates, and the concept of a "mean time" would probably be meaningless. Another bimodally distributed activity might be characterized by one set of time estimates based on the assumption that a closed form solution will be found to a particular problem and another set associated with the need to resort to some time-consuming enumerative technique. Some other activity might involve a series of out-and-try attempts; the manager may prefer to use a multi-modal or a uniform distribution to represent the pdf in this instance.

It is not our intention to develop a number of "standard" distributions for PERT activities, but rather to demonstrate that the requirement that all activity estimates conform to one form of distribution 1) is unnecessary, and 2) can be unduly restrictive, possibly frustrating attempts to apply one's best estimate to an activity pdf. The implementation which is described in Chapter IV accepts pdf's of any form.

CHAPTER III

STATISTICAL DEVELOPMENT

3.1 Procedures for Calculating Event pdf's

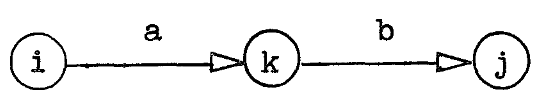
In order to calculate the pdf's associated with each of the events in the network, we will define two operations which may be performed to produce a resultant distribution from two arbitrary input distributions.

1) The pdf's of two activities (or an activity and an event) can be composed in <u>series</u> to yield the pdf of the completion of the two activities performed in sequence.

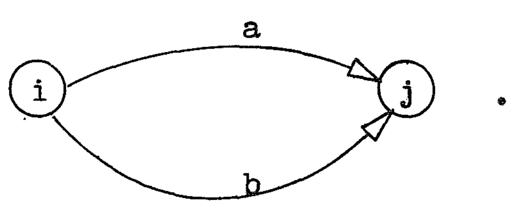
2) The pdf's of two activities (or sequences of activities) can be composed in <u>parallel</u> to obtain the pdf of the time when both activities have been completed if they are performed simultaneously.

We have two basic cases which might be diagramatically represented as follows:

Case 1 (series)



Case 2 (parallel)



In both cases we are given the pdf's for the event i and the activities a and b. As we have indicated in the previous chapters, we can compose the pdf f_i with the composition of f_a and f_b to get f_j.

Further, to denote the "composition process" we will use the symbolism $f_{a \cdot b}$ in the first case and $f_{a + b}$ in the second where these may be called the "serial" and "parallel" composition, respectively. This is the same symbolism that was used in Chapter II, but without formal introduction.

For the algorithms used in the implementation it was found to be more convenient to represent the distributions in the form of the cumulative distribution function (cdf). The cdf of a, denoted F_a, is related to the pdf by

$$F_a(t) = \int_{m_a}^{t} f_a(t_a)dt_a.$$

From the definition of f_a and the restriction that $f_a(t_a) = 0$ for $t_a < m_a$, we observe that $F_a(t)$ is the probability that the duration of activity a is no greater than the interval t and that $F_i(t)$ is the probability that event i will occur at a time no later than t.

<u>Series Composition.</u> If F_a and F_b are composed in series to obtain $F_{a \cdot b}$, the end points of the resultant distribution are

$$m_{a \cdot b} = m_a + m_b ,$$
$$M_{a \cdot b} = M_a + M_b .$$

The probability that $t_a + t_b$ is no greater than t is the sum of the probabilities of all combinations of t_a and t_b which are less than or equal to t,

$pr(t_a+t_b \leq t) = pr(t_a)$ and $pr(t_b \leq t-t_a)$ for all $t_a \leq t-m_b$, or

$$F_{a \cdot b}(t) = \int_{m_a}^{t-m_b} f_a(t_a) \int_{m_b}^{t-t_a} f_b(t_b)dt_b dt_a$$

$$= \int_{m_a}^{t-m_b} f_a(t_a)F_b(t-t_a)dt_a.$$

We can differentiate the expression with respect to t using the method given by Hildebrand ([7], page 360) to demonstrate that it is equivalent to the expression introduced in Chapter II:

$$f_{a \cdot b}(t) = \frac{d}{dt} F_{a \cdot b}(t) = \int_{m_a}^{t-m_b} \left\{ \frac{d}{dt} [f_a(t_a)]F_b(t-t_a) + f_a(t_a)f_b(t-t_a) \right\} dt_a + f_a(m_a)F_b(t-t+m_b) - f_a(t-m_b)F_a(t-m_a) \frac{d}{dt} m_a$$

$$= \int_{m_a}^{t-m_b} f_a(t_a)f_b(t-t_a)dt_a.$$

<u>Parallel Composition.</u> If F_a and F_b are composed in parallel, the end points of the resultant distribution are

$$m_{a+b} = \max(m_a, m_b)$$
$$M_{a+b} = \max(M_a, M_b).$$

The probability that <u>both</u> activities a and b are completed on or before time t is the probability that a is completed on or before time t and that b is completed on or before time t, thus

$$pr(t_a \leq t \text{ and } t_b \leq t) = pr(t_a \leq t) \cdot pr(t_b \leq t), \text{ or}$$

$$F_{a+b}(t) = F_a(t) \cdot F_b(t).$$

Differentiating,

$$f_{a+b}(t) = f_a(t) \cdot F_b(t) + f_b(t) \cdot F_a(t),$$

which is the expression used in Chapter II to perform the parallel composition.

3.2 Combinatorial Techniques of PERT

The series operation used in ordinary PERT analysis is the summation of means and variances from two distributions to obtain these values for the derived distribution; the parallel operation consists of assigning the greatest mean value from among the input distributions and the variance of this distribution as the mean and variance of the derived distribution. We can now investigate the validity of these simplifications.

Mean and Variance of Series Combination. Define the mean of the distribution f_a to be

$$\mu_a = E(t_a) = \int_{m_a}^{M_a} t_a f_a(t_a) dt_a,$$

the mean of the distribution of sums from f_a and f_b will be

$$\mu_{a\cdot b} = E(t_a+t_b) = \int_{m_a}^{M_a}\int_{m_b}^{M_b} (t_a+t_b)\, f(t_a \text{ and } t_b)dt_a dt_b$$

$$= \int_{m_a}^{M_a}\int_{m_b}^{M_b} (t_a+t_b)\, f_a\,(t_a)\, f_b\,(t_b)dt_a\, dt_b$$

$$= \int_{m_a}^{M_a}\int_{m_b}^{M_b} t_a f_a(t_a) f_b(t_b)dt_a dt_b + \int_{m_a}^{M_a}\int_{m_b}^{M_b} t_b f_a(t_a) f_b(t_b) dt_a dt_b$$

$$= \int_{m_a}^{M_a} t_a f_a(t_a)dt_a \int_{m_b}^{M_b} f_b(t_b)dt_b + \int_{m_a}^{M_a} f_a(t_a)dt_a \int_{m_b}^{M_b} t_b f_b(t_b)dt_b.$$

Making use of $\int_{m_a}^{M_a} f_a(t_a)dt_a = 1$, we find

$$\mu_{a\cdot b} = \int_{m_a}^{M_a} t_a\; f_a(t_a dt_a + \int_{m_b}^{M_b} t_b\; f_b(t_b)dt_b$$

$$= \mu_a + \mu_b\,.$$

So the mean of the distribution of the sum is equal to the sum of the means of the individual distributions, as is assumed in PERT.

To find the variance of this sum we define

$$\sigma_a^{\,2} = \int_{m_a}^{M_a} (t_a - \mu_a)^2 f_a(t_a)\,dt_a$$

so that,

$$\sigma_{a\cdot b}^{\,2} = \int_{m_a}^{M_a} \int_{m_b}^{M_b} (t_a + t_b - \mu_{ab})^2 \, f(t_a \text{ and } t_b)\,dt_a dt_b$$

$$= \int_{m_a}^{M_a} \int_{m_b}^{M_b} (t_a + t_b - \mu_a - \mu_b)^2 f_a(t_a) f_b(t_b)\,dt_a dt_b$$

$$= \int_{m_a}^{M_a} t_a^{\,2} f_a(t_a)\,dt_a \int_{m_b}^{M_b} f_b(t_b)\,dt_b + 2\int_{m_a}^{M_a} t_a f_a(t_a)\,dt_a \int_{m_b}^{M_b} t_b f_b(t_b)\,dt_b$$

$$+ \int_{m_a}^{M_a} f_a(t_a)\,dt_a \int_{m_b}^{M_b} t_b^{\,2} f_b(t_b)\,dt_b - 2\mu_a \int_{m_a}^{M_a} f_a(t_a)\,dt_a \int_{m_b}^{M_b} t_b f_b(t_b)\,dt_b$$

$$- 2\mu_b \int_{m_a}^{M_a} t_a f_a(t_a)\,dt_a \int_{m_b}^{M_b} f_b(t_b)\,dt_b - 2\mu_a \int_{m_a}^{M_a} f_a(t_a)\,dt_a \int_{m_b}^{M_b} t_b f_b(t_b)\,dt_b$$

$$\sigma_{a,b}^2 \text{ (cont'd.)} \quad - 2\mu_b \int_{m_a}^{M_a} f_a(t_a)dt_a \int_{m_b}^{M_b} t_b f_b(t_b)dt_b$$

$$+ \mu_a^2 \int_{m_a}^{M_a} f_a(t_a)dt_a \int_{m_b}^{M_b} f_b(t_b)dt_b$$

$$+ 2\mu_a\mu_b \int_{m_a}^{M_a} f_a(t_a)dt_a \int_{m_b}^{M_b} f_b(t_b)dt_b$$

$$+ \mu_b^2 \int_{m_a}^{M_a} f_a(t_a)dt_a \int_{m_b}^{M_b} f_b(t_b)dt_b$$

$$= \int_{m_a}^{M_a} t_a^2 f_a(t_a)dt_a + 2\mu_a\mu_b + \int_{m_b}^{M_b} t_b^2 f_b(t_b)dt_b$$

$$- \int_{m_a}^{M_a} 2\mu_a t_a f_a(t_a)dt_a - 2\mu_a\mu_b - 2\mu_a\mu_b$$

$$- \int_{m_b}^{M_b} 2\mu_b t_b f_b(t_b)dt_b + \int_{m_a}^{M_a} \mu_a^2 f_a(t_a)dt_a + 2\mu_a\mu_b$$

$$+ \int_{m_b}^{M_b} \mu_b^2 f_b(t_b)dt_b$$

$$= \int_{m_a}^{M_a} (t_a^2 - 2\mu_a t_a + \mu_a^2) f_a(t_a)dt_a + \int_{m_b}^{M_b} (t_b^2 - 2\mu_b t_b + \mu_b^2) f_b(t_b)dt_b$$

$$= \int_{m_a}^{M_a} (t_a - \mu_a)^2 f_a(t_a)dt_a + \int_{m_b}^{M_b} (t_b - \mu_b)^2 f_b(t_b)dt_b$$

$$= \sigma_a^2 + \sigma_b^2 .$$

These results demonstrate that the method used in PERT of summing means and variances along a path is quite valid, and if there were no parallel paths in a network, the PERT analyst would be completely justified in discarding the activity distributions and using only their means and variances to calculate the expected event times. It is in the parallel composition that errors are introduced.

Mean of a Parallel Composition. As has been noted previously, the time of occurrence of an event j is the latest time of completion of the activities terminated by event j, and the mean time of completion of both activities a and b is $\mu_{a+b} = E\{\max(t_a, t_b)\}$. To evaluate this sum, let us make the substitution

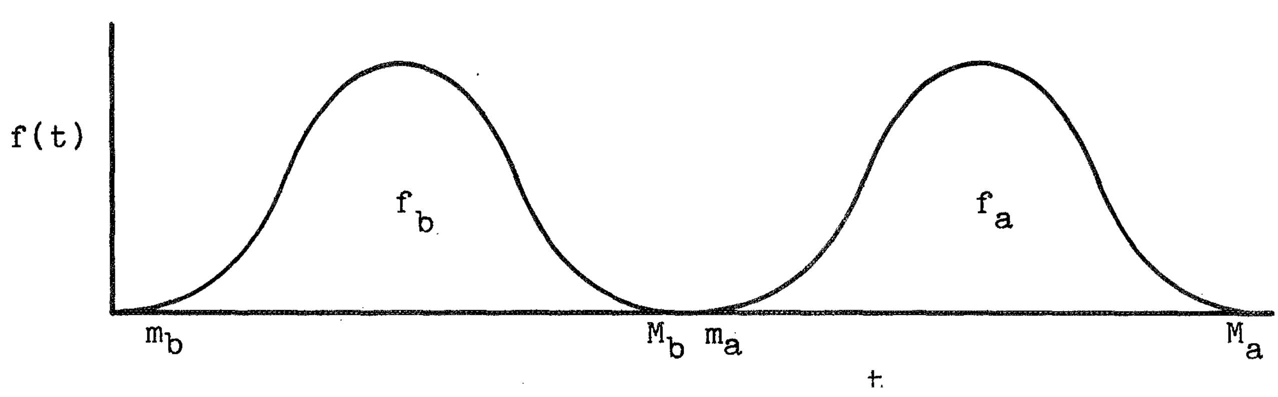
$$\max(t_a, t_b) = \tfrac{1}{2}(t_a + t_b + |t_a - t_b|).$$

This may be proved by considering first the case of $t_a \geq t_b$, in which the right hand side reduces to t_a, and the case of $t_b > t_a$, in which it reduces to t_b. Using the result derived above for the expected value of a sum, we find that

$$E[\max(t_a, t_b)] = \tfrac{1}{2}[\mu_a + \mu_b + E(|t_a - t_b|)] .$$

So, to determine μ_{a+b}, we must evaluate $E(|t_a - t_b|)$.

Let us first assume that the distributions f_a and f_b do not overlap, because $m_a > M_b$. Pictorially,



Then $E(|t_a - t_b|)$ becomes $E(t_a - t_b)$, which is $\mu_a - \mu_b$, so that

$$\mu_{a+b} = \mu_a .$$

So here the standard PERT assumption that the composite mean of the two paths in parallel is equal to the mean value of the longer path introduces no errors in determining μ_{a+b}.

If, however, the distributions do overlap, the situation changes. Let us assume that $f_b = f_a$ is normal [8] with a mean μ_a and variance σ_a^2. As a first step in the evaluation of $E(|t_a - t_b|)$, we note that $f(t_a - t_b)$ is normal with a mean of 0 and a variance of $2\sigma_a^2$,

$$f(t_a - t_b) = \frac{1}{2\sigma_a\sqrt{\pi}} \exp\left[-\frac{(t_a - t_b)^2}{4\sigma_a^2}\right].$$

Also,

$$f(|t_a - t_b|) = f(t_a - t_b) + f(t_b - t_a)$$

$$= \frac{1}{2\sigma_a\sqrt{\pi}} \exp\left[-\frac{(t_a - t_b)^2}{4\sigma_a^2}\right] + \frac{1}{2\sigma_a\sqrt{\pi}} \exp\left[-\frac{(t_b - t_a)^2}{4\sigma_a^2}\right]$$

$$= \frac{1}{\sigma_a\sqrt{\pi}} \exp\left[-\frac{(t_a - t_b)^2}{4\sigma_a^2}\right] .$$

Let $X = |t_a - t_b|$. Then,

$$f(X) = \begin{cases} \frac{1}{\sigma_a\sqrt{\pi}} \exp\left[- X^2/4\sigma_a^2\right], & X \geq 0 \\ 0 , & X < 0. \end{cases}$$

8/ f_a can be made very nearly normal, so long as we retain $m_a \geq 0$ and M_a finite. A good approximation is obtained if we let $f_a(t) = n(\mu_a, \sigma_a)$ for $\mu_a - 3\sigma_a \leq t \leq \mu_a + 3\sigma_a$, $f_a(t) = 0$ elsewhere.

The expected value of X is then

$$E(X) = \frac{1}{\sigma_a\sqrt{\pi}} \int_0^\infty X \exp\left[- X^2/4\sigma_a^2\right] dX$$

$$= - \frac{2\sigma_a}{\sqrt{\pi}} \int_0^\infty - \frac{X}{2\sigma_a^2} \exp\left[- X^2/4\sigma_a^2\right] dX$$

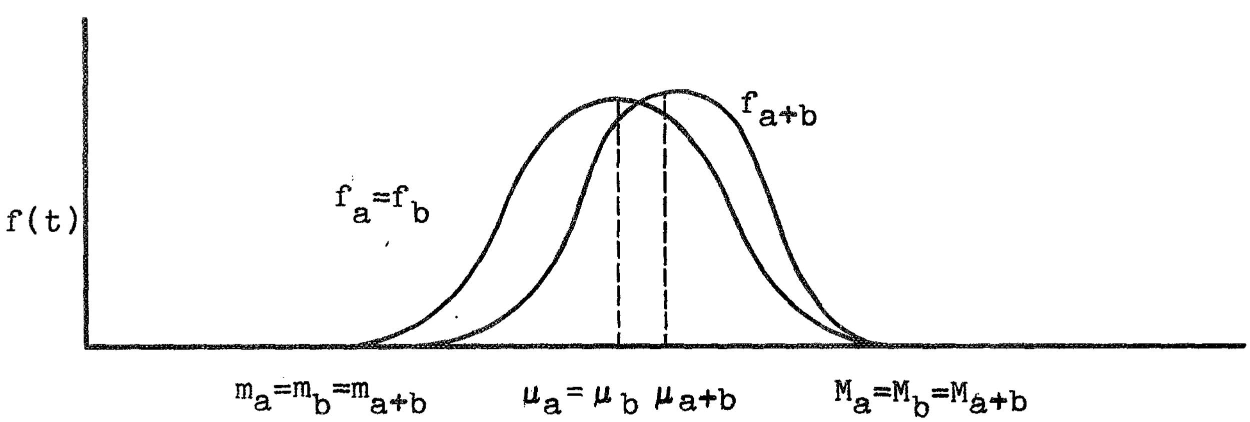
$$= - \frac{2\sigma_a^2}{\sqrt{\pi}} \exp\left[X^2/4\sigma_a^2\right]\Big|_0^\infty$$

$$= \frac{2\sigma_a}{\sqrt{\pi}} .$$

So we find that

$$\mu_{a+b} = \mu_a + \sigma_a/\sqrt{\pi} \approx \mu_a + .56\ \sigma_a .$$

In this case of normal distributions, the approximation $\mu_{a+b} \approx \max(\mu_a, \mu_b)$, which is ordinarily used in PERT analysis, introduces an error slightly greater than one-half a standard deviation. This result can be represented pictorially,



This result could have been obtained directly from the cdf tables found in Dixon and Massey [6]. The μ_{a+b} is that value of t for which $F_a(t) \cdot F_b(t) = 0.5$. We know that $F_a = F_b$, so $F_a(t) = \sqrt{0.5} = 0.71$. Using the cdf table for the normal distribution, we find that $F_a(t) = 0.71$ for $t = \mu_a + 0.56\sigma_a$.

When f_a and f_b are arbitrary pdf's, there is no convenient closed form representation of μ_{a+b} or σ_{a+b} (see, for example, Chapter II where both pdf's are known to be uniform). However, these calculations with normal distributions, together with the results of Chapter II show that the standard PERT assumptions to the effect that

$$\mu_{a+b} \approx \max(\mu_a, \mu_b)$$

and

$$\sigma_{a+b} \approx \sigma \max(\mu_a, \mu_b)$$

can be quite misleading. The only accurate method is to return to the original definition and evaluate μ_{a+b} as

$$\mu_{a+b} = t_{a+b} \text{ where } t_{a+b} \text{ satisfies } F_{a+b}(t_{a+b}) = 0.5.$$

Evaluation of the F_{a+b} for many points t given this open form is usually impractical by hand but relatively easy by computer if one approximates the integral by finite summation techniques. However, in order to calculate μ_{a+b} we must now have the actual distributions F_a and F_b (or f_a and f_b) readily available. This is our major difference from standard PERT implementations which retain only μ_a and σ_a as approximate distribution parameters.

Given F_a and F_b, the computer procedure to calculate F_{a+b} or $F_{a \cdot b}$ is straightforward. Figures 11, 12, 13, and 14 which were previously presented to support our contention that complete distributions must be retained in the data structure for analysis of a PERT network, were generated by this procedure. So, too, was Figure 15 which supports our last calculation of μ_{a+b} where $f_a = f_b$ is a nearly normal distribution. For this plot, the input distributions have a mean $\mu_a = 12.0$ and standard deviation $\sigma_a = 1.0$. The non-zero range of these distributions is from $m_a = \mu_a - 3\sigma_a = 9.0$ to $M_a = \mu_a + 3\sigma_a = 15.0$. f_a and f_b are combined in parallel to obtain f_{a+b}. While f_{a+b} (9.0) may be assumed to have a non-zero value, no appreciable value for $f_{a+b}(t)$ is seen for any $t < 10.2$. This plot demonstrates that the mean of the resultant $\mu_{a+b} > \mu_a$. The routine which generated the plot also calculated $\mu_{a+b} = 12.56$, which is in agreement with our analytically calculated value, and $\sigma_{a+b} = 0.82$.

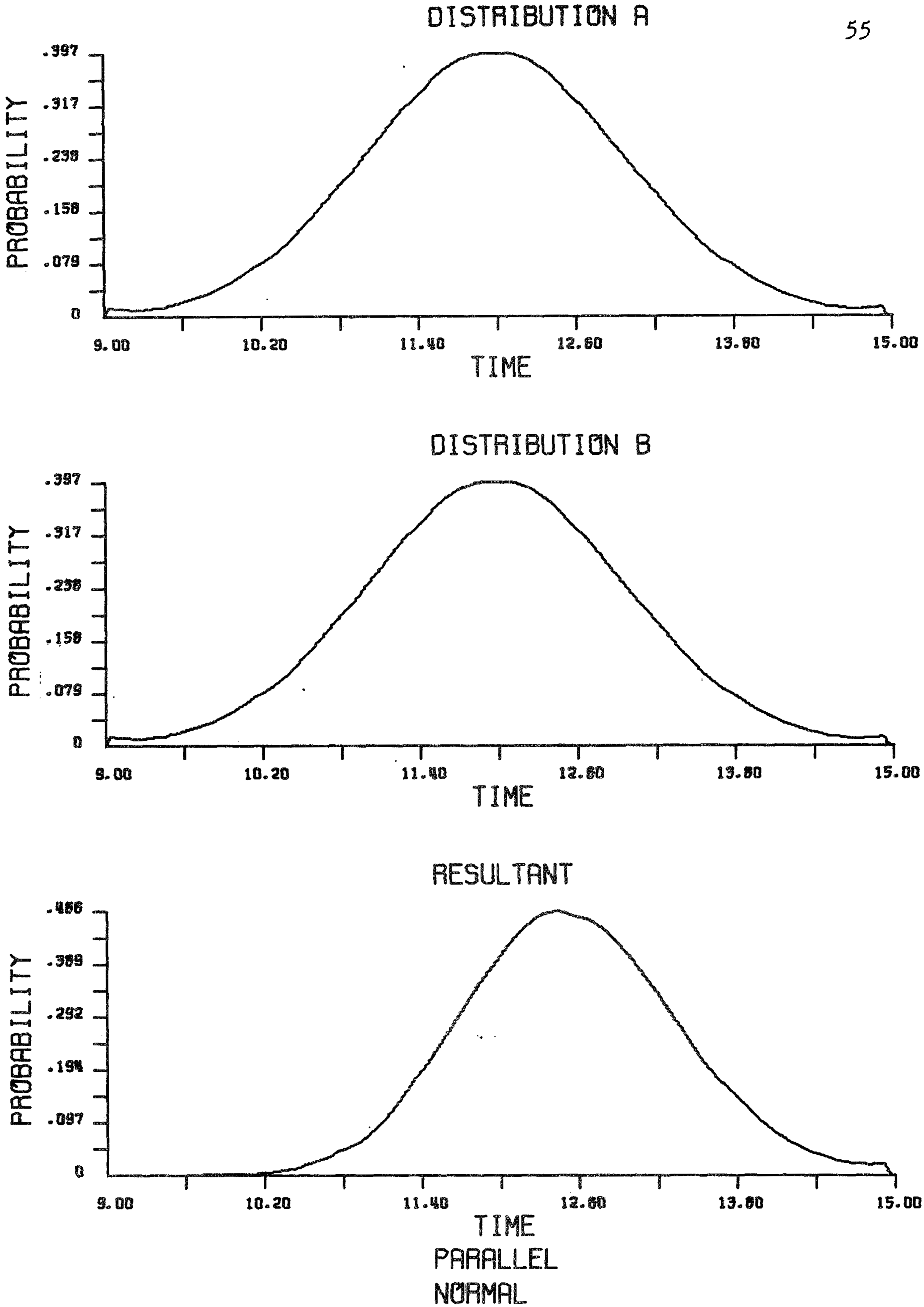


Figure 15. Parallel Combinations of Two Nearly-Normal Distributions

CHAPTER IV

IMPLEMENTATION

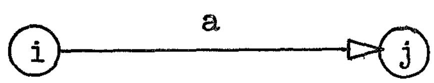
4.1 Basic Data Structure

The computer model which is described here demonstrates a methodology for determining the pdf associated with any of the events in the project, and in particular that of project completion, once the PERT network has been established and a pdf defined for each of the activities. This model has been implemented on the UNIVAC 1108 at the University of Maryland, using the RSVP list processing language [8], developed by Robert Liebermann.

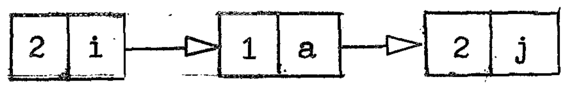
The basic element of RSVP is called an <u>atom</u>. RSVP atoms may be used to represent different types of elements within a data structure by assigning to each atom an integer called its <u>type</u>. The elements of a PERT network are the activities (arcs) and events (points); activities are represented by type 1 atoms and events are initially represented by type 2 atoms. As information becomes known about the pdf associated with an event, its atom type will be varied. In the accompanying figures, an atom will be pictured as

<type>	<name>

where <type> is the integer identifying the type of element (activity, event, etc.) represented by the atom and <name> is the symbol (a, b, c,..., for activities; 1, 2, 3, ..., for events) identifying the particular element. In the following discussion atoms will be referred to by name; e.g., atom 30 is the atom representing event 30. Pointers are established linking the appropriate activities and events; the following subnetwork



is represented by a substructure whose appearance is



.

In addition, we can associate with any atom any collection of data. Since the atoms of our data structure correspond to the points and edges of the PERT network, we will call such associated data the labels of the atom. This terminology agrees with that of graph theory, where one speaks of labeled edges. It should not be confused with the common computer usage in which a label is treated as synonymous with identifier. Note that just as one can associate more than one label with the points and/or edges of a graph, so we may have several labels, or collections of data, associated with the atoms of our data structure.

The preceding chapters have established that, for accurate PERT analysis, the correct labels to assign to the events and activities of the network are their actual probability distributions, not a condensed version consisting of the mean and variance alone. So these distributions are the data that we associate to the RSVP atoms in our computer implementation.

Initially, pdf's are known for only the activities and event 0. These pdf's could be stored as labels; however, we actually store cdf's because the are more convenient to use in the operations for combining distribution functions in parallel and in series. For continuous functions, as are assumed here, a one-to-one correspondence exists between the pdf and cdf, so we may construct a distribution of the one form from the other, as necessary.

An individual F_a is stored as a 13 place vector containing $\{m_a, M_a, F_a(t_0), \ldots, F_a(t_{10})\}$, where $t_n = m_a + n \dfrac{M_a - m_a}{10}$. We are then approximating a continuous cdf with 10 points, interpolating with quadratic functions between the points. 9/

4.2 Calculation and Assignment of Event Distributions

The process of determining the parallel or series composition of the two distributions F_a and F_b consists first of finding the points t which $F_{ab}(t)$ will be evaluated. As we saw in Chapter III,

$$m_{a+b} = \max(m_a, m_b)$$
$$M_{a+b} = \max(M_a, M_b)$$
$$m_{a\cdot b} = m_a + m_b$$
$$M_{a\cdot b} = M_a + M_b\ .$$

9/ The choice of 10 points to represent the cdf is purely arbitrary, a tradeoff between the accuracy with which the cdf is approximated and timing and storage requirements. Because some cdf's may have a wide range, but with F(t) highly variable over a small range of t, it may be preferable in some applications to vary the time interval between points.

For each point t, F(t) must be evaluated. For the parallel composition,

$$F_{a+b}(t) = F_a(t) \cdot F_b(t).$$

In the series composition, as we noted in the previous chapter,

$$F_{a \cdot b}(t) = \int_{t_a < t-m_b} f_a(t_a)\, F_b(t-t_a)$$

We will approximate this integral with the finite sum resulting from replacing $f_a(t_a)$ with $\dfrac{F_a(t_a+\Delta t_a/2)-F_a(t_a-\Delta t_a/2)}{\Delta t_a}$ where $\Delta t_a = \dfrac{M_a - m_a}{10}$, and substituting Δt_a for dt_a, to obtain

$$F_{a \cdot b}(t) = \sum_{t_a < t-m_b} [F_a(t_a+\Delta t_a/2)-F_a(t_a-\Delta t_a/2)]\, F_b(t-t_a).$$

Input to the model consists of information about the individual activities; for each, the initiating and terminating events and the cdf are specified. The existence of an event is implied by its appearance in the activity information, and an atom representing an event is created upon the first reference to the event in the input. For each activity, an activity atom is created and linked to the atoms of the bracketing events, and the cdf is stored and linked to the activity atom. Figure 16 shows a PERT network whose activities have been labeled with pdf's and a corresponding structure as it appears in storage.

In Figure 16, all of the event atoms are type 2, with the exception of atom 0 which is designated type 3. This

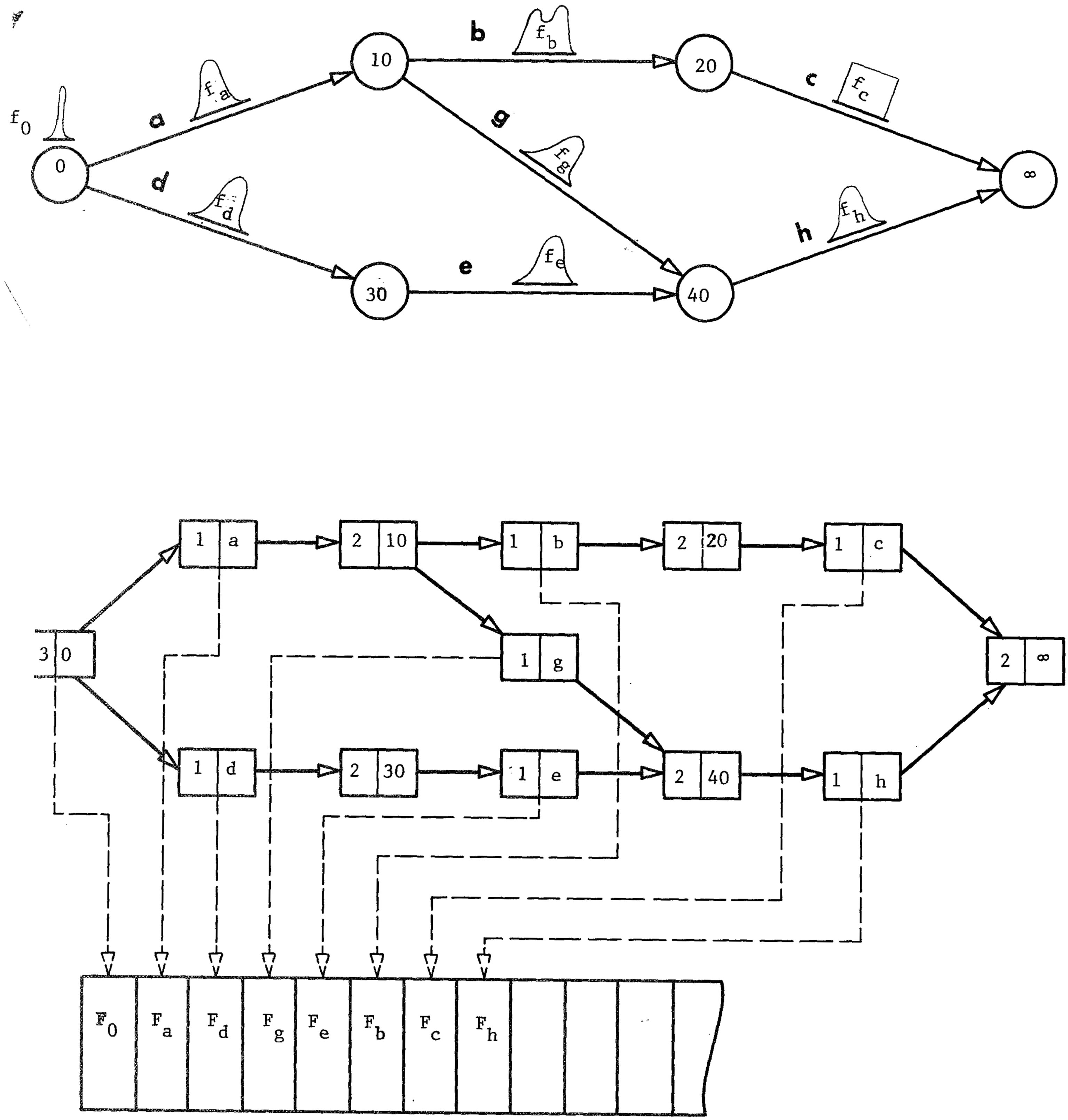
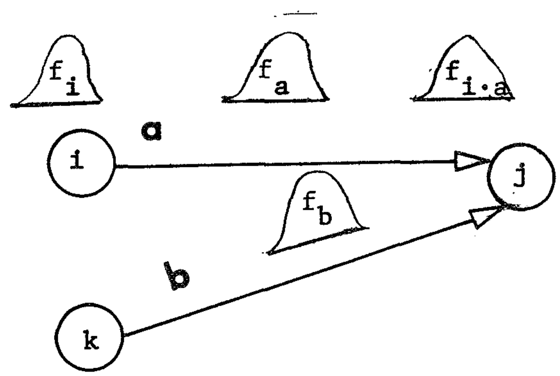


Figure 16. Initial PERT Network and Storage Equivalent

type of atom will be used to represent an event whose cdf has been determined. Event 0 serves as a starting point for a recursive procedure for calculating the cdf's of all events in the network. Any known F_i is used as the distribution of starting times for each activity a initiated by event i. F_i is composed in series with each F_a to yield $F_{i \cdot a}$ which is used in determining the cdf of the event terminating a. Let us assume that the particular activity a is terminated by event j. $F_{i \cdot a}$ is the distribution of the time of <u>completion</u> of activity a (as distinguished from the input F_a, which is the distribution of the <u>duration</u> of a). If no other activity terminates at j, then $F_j = F_{i \cdot a}$. If, however, another activity b is initiated by an event k, pictorially,



then to determine F_j, we must compose $F_{i \cdot a}$ in parallel with $F_{k \cdot b}$. Until $F_{k \cdot b}$ has been calculated, we will store $F_{a \cdot i}$ as a temporary label for event j, denoted F'_j and called a "partial distribution" of j. After the cdf's of all activities terminated by j have been used in the calculation of F'_j, it is denoted F_j and referred to as the <u>forward</u> <u>distribution</u> of j.

This process will now be illustrated for the PERT network pictured in Figure 16. Activities a and d are initiated at time 0, so F'_{10} is set equal to F_a and F'_{30} is set equal to F_d. Because there is only one path to each of these activities, the cdf's are renamed F_{10} and F_{30} and their atoms become type 3. Both of the events are placed in a "next event" list, a list which points to the events available for use in the series operation.

Now some event, such as 10, is chosen from the "next event" list. F_{10} and F_b in series determine the distribution F_{20}; atom 20 is changed to type 3 and event 20 is added to the "next event" list. F_{10} and F_g are combined in series to yield an F'_{40}. Figure 17 reflect the processing which has been performed to this point.

Next, F_{30} and F_e are combined in series, and the resultant is combined in parallel with F'_{40} to obtain F_{40}. Series combination of F_{20} and F_c gives an F'_∞ ; the resultant of the series combination of f_{40} and f_h is combined in parallel with F'_∞ to determine F_∞ . In deriving F_∞ this implementation already presents more information than a simple PERT analysis using the assumption of a critical path. Given the actual distribution F_∞, the project manager is able to more realistically choose a project completion date t_∞ , from which the necessary times of the events can be derived. For example, a conservative manager may choose for the project completion date t_∞ a time somewhat later than μ_∞ depending upon the shape of F_∞ . On the other hand, if the payoff

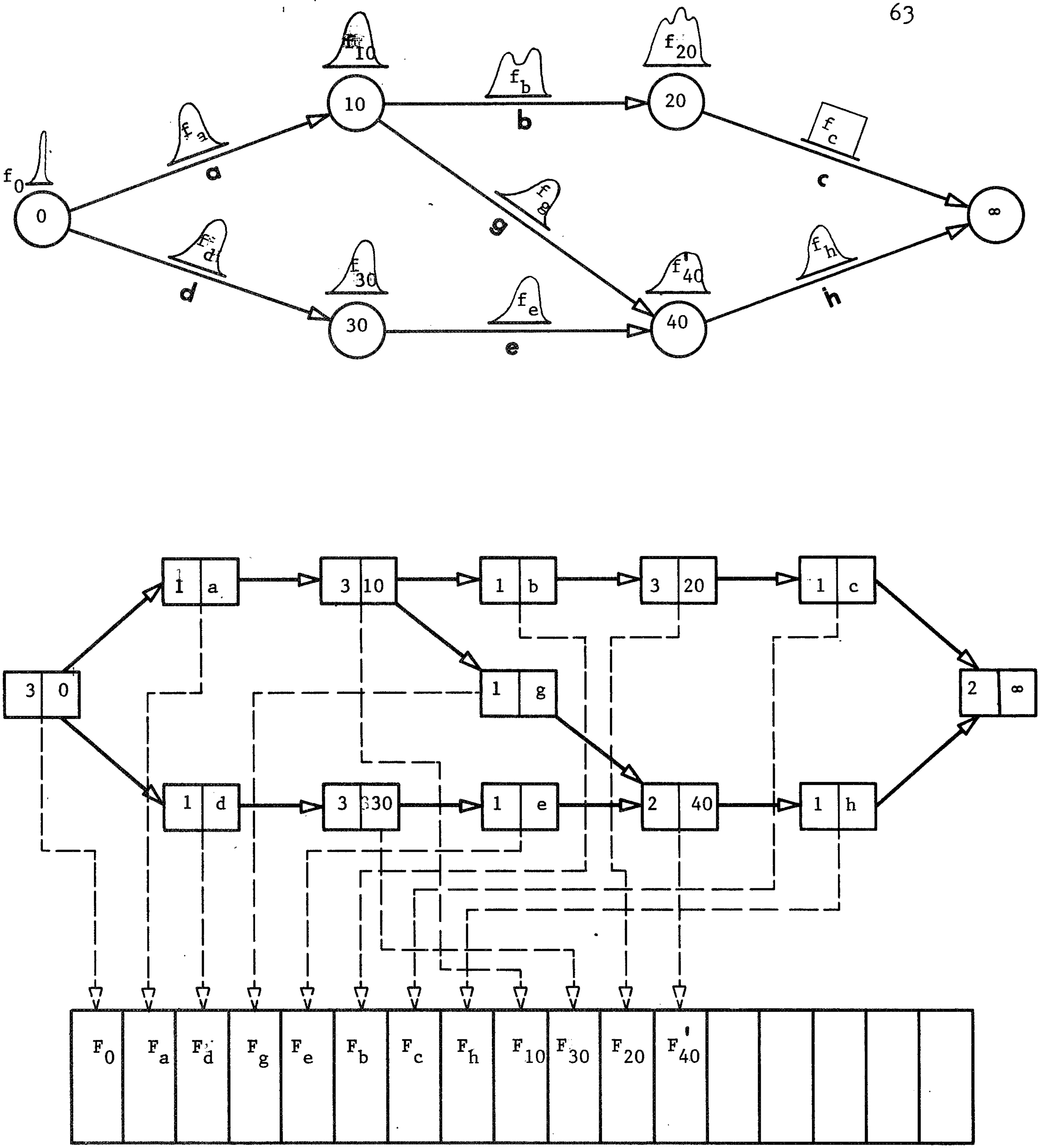


Figure 17. PERT Structure After the Calculation of f_{20}.

appeared sufficient, he might gamble on an earlier date, but at least he would be apprised of the risk involved. Lacking interactive capabilities, the implementation described here automatically assigns μ_∞ as t_∞ .

The time required to complete all activities subsequent to an event i is a function of the activity cdf's rather than a fixed time, and therefore the latest time for an event to occur and permit the project to remain on schedule is a cdf, denoted G_i, and called the <u>backward distribution</u> of i. This cdf replaces the single value LET_i in PERT notation. G_i is obtained by combining in parallel the cdf's of all paths (i, ∞) and subtracting the time values of the resulting cdf from t_∞.

Returning to the example, G_{20} is found by subtracting F_c from t_∞ and G_{40} by substracting F_h from t_∞. Atoms 20 and 40 are now designated type 4, indicating that both forward and backward distributions have been calculated, and events 20 and 40 are placed in the "next event" list. A step from event 40 yields G_{30} and a G'_{10}; G_{20} and F_b are composed in series to yield a cdf which is composed in parallel with G'_{10} to determine G_{10}. It would be pointless to specify a G_0, since all times are measured relative to t_0, and so this backward pass is completed. Figure 18 displays the PERT network after all activities and event have been labeled with their cdf's.

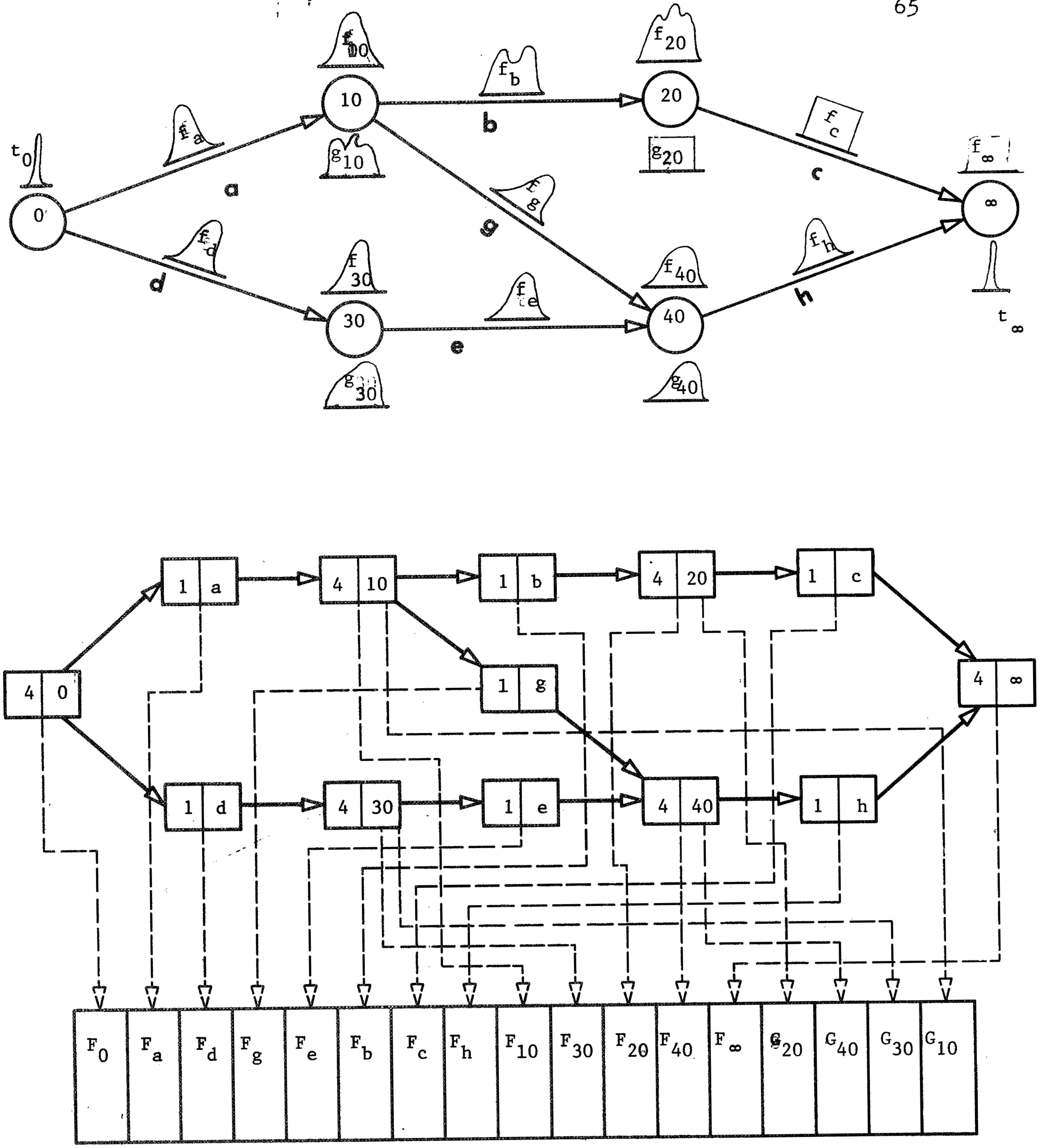


Figure 18. PERT Structure With All pdf's Shown

4.3 Storage Requirements

The RSVP system allocates five words of storage for each atom it creates. This five word area is known as the atom head and contains such information as the atom number (a unique identifier assigned by RSVP), location of the inward and outward links (which will be explained below), and a pointer to the data for the atom.

Each event in the network is represented by a single atom and in addition requires 13 words for its cdf. If both forward and backward cdf's were retained, as would be the case in an interactive system which allowed the project manager to retrieve the forward and backward distribution or the slack of any event in the network, 31 words of storage would be required for each event in the network. However, in this implementation we keep only one distribution at a time. In addition, four words are used to store the mean and variance of the forward and backward distributions. Two additional words are used per event for a list which links (PERT) event number to (RSVP) atom number. In this implementation, then, the storage required for Ne events is Ne (5 + 13 + 4 + 2) = 24 Ne words.

Each activity also requires five words for its atom head and thirteen for its cdf. In addition, each activity is linked to two events. Linkage is accomplished in RSVP through use of outrings and inrings, which are lists of pointers (links) to and from other atoms. For each activity, there will be an out link attached to the atom of the initiating event, an in link and an out link for the activity atom, and an in link for the terminating event. Na activities, therefore, require Na (5 + 13 + 4) = 22 Na words of core.

The total storage requirement is 23 Ne + 22 Na words, If the ratio of activities to events in the network is about 3 to 2, a relatively large 500 event network would require about 28,500 words. This number might be considered to be near the limit of the "fast" space available, although the virtual memory capability of RSVP makes the total space available practically limitless.

List processors that operate in a virtual memory space map (or page) the excess over fast core capacity to some peripheral device such as tapes, disk, or drum. In general, the resultant access or page swapping can be expected to significantly degrade execution time. However, if we consider an actual PERT network we see that most processing is local in the sense that only a small portion of the entire network is necessary for processing at any single time. There is no need, for example, to retain in core all those activities or events for which the forward (or backward) pdf has been calculated and which are not involved in current calculations.

The auxilliary storage facility of RSVP was designed to take advantage of just such situations. Atoms are not paged, but treated on an individual basis. Individual atoms are not mapped into fast core until needed (i.e., referenced) and they are individually mapped out on the basis of usage. Consequently, at any time we may expect that the data structure in fast memory closely corresponds to the actual collection of events and activities we are then working with, and that only one access to peripheral storage will be required per event or activity for any given pass through the network.

Consequently, core requirements might be effectively reduced to about 2,000 words (for any size network) without unduly degrading performance.

4.4 Execution Times

Once RSVP space has been initialized and the PERT network built, the time require to solve the network is approximately proportional to the number of activities in the network. For each activity, the cdf of time of completion is found by taking the serial composition of the activity cdf and the cdf of the initiating event. This composite cdf is then assigned to the terminating event or, if a cdf has been previously assigned to the terminating event, the two cdf's are composed in parallel. Therefore, for each event, the number of parallel operations performed is one less than the inward degree of the event. The total inward degree of the network is equal to the number of activities. The total number of parallel operations performed in calculating forward distributions is Na - (Ne - 1), because the inward degree of event 0 is 0. Similar analysis yields Na series and Na - Ne + 1 parallel operations in calculating the backward distributions. Total time consumed in solving the network is then:

$$T = 2[NaTs + (Na - Ne + 1)\ Tp],$$

where Ts is the time required for a series composition and Tp is the time required for a parallel composition.

Chapter V presents examples of output from this model. The total time required for an 11 event 15 activity network, including RSVP initialization, model input, network solution and output was 2.9 seconds.

CHAPTER V

SAMPLE OUTPUT

The model which we described in the previous chapter has been exercised to demonstrate the calculation of event distributions in a small PERT network. The results of these runs are presented in Figures 19 through 21. For simplicity, all of the activity distributions are of the nearly normal type introduced in the examples in Chapter III. For each activity a, given μ_a and σ_a, the end points of the cdf are $m_a = \mu_a - 3\sigma_a$ and $M_a = \mu_a + 3\sigma_a$.

In these illustrations, the input to the model is shown under the heading "ACTIVITY DISTRIBUTIONS". The output is listed under "EVENT DISTRIBUTIONS". The network is displayed with the activities labeled with their distribution means and variances.

In the input listing, the bracketing events of each activity are shown under "INITIATING EVENT" and "TERMINATING EVENT". The distribution parameters μ_a and σ_a^2, are not input; instead, the 13 place vector described in Chapter IV is used to specify F_a. The model calculates μ_a and σ_a^2 and lists them under "DURATION", "MEAN", and "VARIANCE". Finally, F_a as input (in vector form) is listed under "INPUT DISTRIBUTION".

Output from the model ("EVENT DISTRIBUTIONS") consists of the derived information about each event, listed under "EVENT" in order of increasing expected time μ_i. The mean and variance σ_i^2 of F_i are listed under "FORWARD DISTRIBUTION", followed by the mean and variance of G_i under "BACKWARD DISTRIBUTION". The "SLACK" listed is the difference between the mean of G_i and that of F_i, and may be thought of as the mean of slack distribution. The concept of a slack distribution will be considered in greater detail in the following chapter.

In performing ordinary PERT analysis on the network shown in Figure 19, we would note that the longest path is $\rho(0,20,50,80,100)$, with $L_\mu(\rho) = 28.0$, and applying the critical path assumption, we would assign $\mu_{100} = 28.0$ (in this implementation event has been numbered event 100). However, this network contains another path, $\rho(0, 10, 40, 70, 100)$, on which $L_M(\rho)$ is 37.0. As we have seen, the presence of a second path whose distribution overlaps that of the "critical" path (we might call this a "near-critical" path) tends to shift the distribution of predicted project completion time toward a later date, and for this particular network the shift in μ_{100} is about 0.5; the model calculates μ_{100} to be about 28.5.

An unexpected consequence of using individual activity pdf's to calculate the event pdf's is revealed in the column listing slack. We recall that one characteristic of a critical path is that all events on that path contain zero slack. In this case, however, with the expected duration of the entire project exceeding that of any individual path, all events are shown to have some slack. This means that any

ACTIVITY DISTRIBUTIONS

INITIATING EVENT	TERMINATING EVENT	DURATION MEAN	DURATION VARIANCE	INPUT DISTRIBUTION m_a	INPUT DISTRIBUTION M_a
0	10	3.000	.441	1.000	5.000
0	20	6.000	.441	4.000	8.000
0	30	5.000	.110	4.000	6.000
10	50	2.000	.028	1.500	2.500
10	40	10.000	.991	7.000	13.000
20	50	5.000	.110	4.000	6.000
30	60	3.000	.028	2.500	3.500
30	90	5.000	.110	4.000	6.000
40	70	9.000	.441	7.000	11.000
50	80	9.000	.991	6.000	12.000
60	80	3.001	.108	2.000	4.000
60	90	2.001	.027	1.500	2.500
70	100	5.001	.108	4.000	6.000
80	100	8.004	.970	5.000	11.000
90	100	4.001	.108	3.000	5.000

EVENT DISTRIBUTIONS

EVENT	FORWARD MEAN	DISTRIBUTION VARIANCE	BACKWARD MEAN	DISTRIBUTION VARIANCE	SLACK
0	.000	.000	.000	.000	.000
10	3.000	.441	4.483	1.558	1.483
30	5.000	.110	14.482	1.305	9.482
20	6.000	.441	6.484	2.346	.484
60	8.000	.140	17.482	1.101	9.482
90	10.253	.135	24.488	.108	14.235
50	11.000	.561	11.484	1.982	.484
40	13.000	1.441	14.488	.551	1.488
80	20.000	1.579	20.485	.970	.485
70	22.000	1.960	23.488	.108	1.488
100	28.489	1.849	28.489	.000	.000

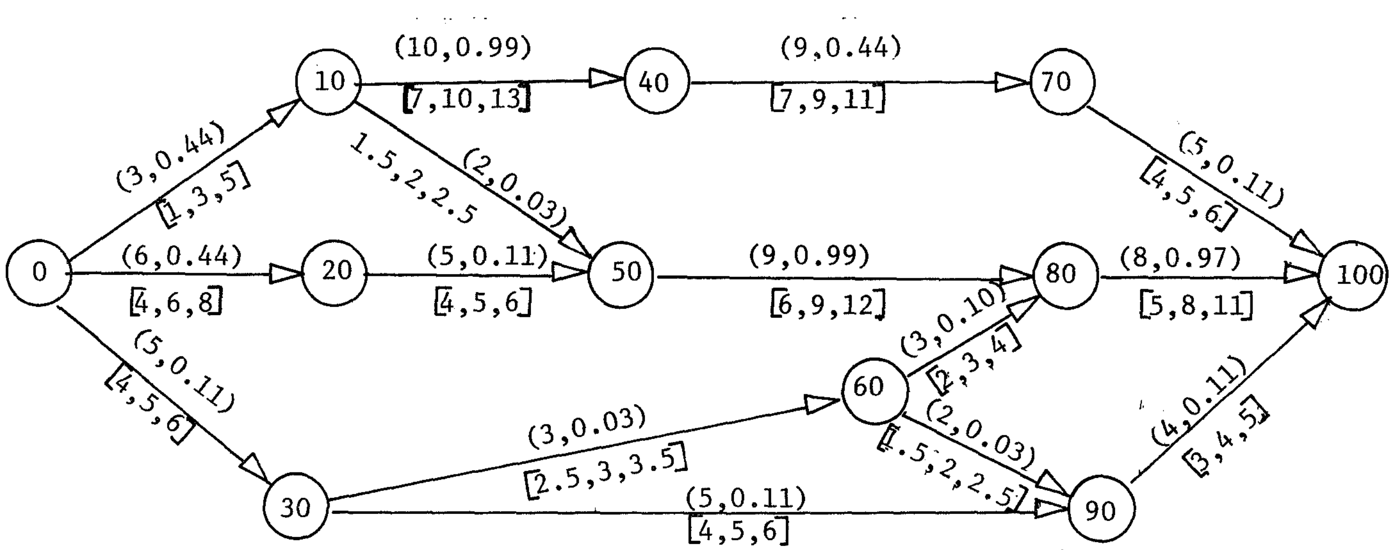


Figure 19. Sample Model Output

sequence of activities in the project can consume more than its predicted time without causing project slippage. No individual sequence is absolutely "critical".

We noted previously that for networks containing no "near-critical" paths, the critical path assumption holds and project completion is indeed determined by only one path. On the other hand, as the lengths of other paths approach that of the longest path the duration of the project surpasses that of any path. We can observe the sensitivity of project duration to changes in the length of a "near-critical" path by varying the cdf of the activity linking event 40 to event 100. In Figure 20, M_a has been increased so that μ_a of the activity is increased from 9.0 to 9.9. The length of the longest path is unchanged from the previous example, so the PERT predicted project length is still 28.0; however, the expected duration calculated by the model is now 28.9. An increase of 0.9 in a near-critical path has resulted in an increase of about 0.4 in the overall project duration.

M_a of this same activity is again lengthened, so that in Figure 21 its mean is 10.0. Now $L\mu$ of (0, 10, 40, 70, 100) is 28.0, so there are two "critical" paths. As before, the increase in the overall project length (from 28.90 to 28.96) calculated by the model is about one-half as great as the increase in the mean length of a near-critical path. If

ACTIVITY DISTRIBUTIONS

INITIATING EVENT	TERMINATING EVENT	DURATION MEAN	DURATION VARIANCE	INPUT DISTRIBUTION m_a	INPUT DISTRIBUTION M_a
0	10	3.000	.441	1.000	5.000
0	20	6.000	.441	4.000	8.000
0	30	5.000	.110	4.000	6.000
10	50	2.000	.028	1.500	2.500
10	40	10.000	.991	7.000	13.000
20	50	5.000	.110	4.000	6.000
30	60	3.000	.028	2.500	3.500
30	90	5.000	.110	4.000	6.000
40	70	9.900	.926	7.000	12.800
50	80	9.000	.991	6.000	12.000
60	80	3.000	.110	2.000	4.000
60	90	2.000	.028	1.500	2.500
70	100	5.000	.110	4.000	6.000
80	100	8.000	.991	5.000	11.000
90	100	4.000	.110	3.000	5.000

EVENT DISTRIBUTIONS

EVENT	FORWARD DISTRIBUTION MEAN	FORWARD DISTRIBUTION VARIANCE	BACKWARD DISTRIBUTION MEAN	BACKWARD DISTRIBUTION VARIANCE	SLACK
0	.000	.000	.000	.000	.000
10	3.000	.441	3.998	2.064	.998
30	5.000	.110	14.900	1.333	9.900
20	6.000	.441	6.900	2.370	.900
60	8.000	.140	17.900	1.130	9.900
90	10.252	.135	24.900	.110	14.648
50	11.000	.561	11.900	2.006	.900
40	13.000	1.441	14.000	1.042	1.000
80	20.000	1.579	20.900	.991	.900
70	22.900	2.447	23.900	.110	1.000
100	28.900	1.925	28.900	.000	.000

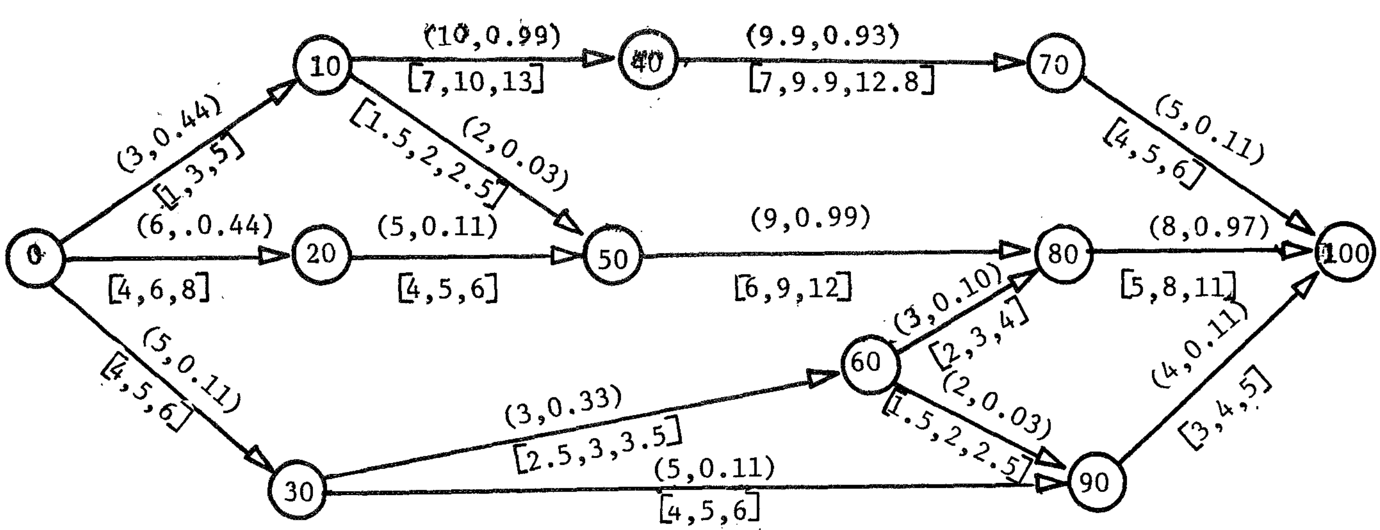


Figure 20. Network After Lengthening a Near-Critical Path

ACTIVITY DISTRIBUTIONS

INITIATING EVENT	TERMINATING EVENT	DURATION MEAN	DURATION VARIANCE	INPUT DISTRIBUTION m_a	INPUT DISTRIBUTION M_a
0	10	3.000	.441	1.000	5.000
0	20	6.000	.441	4.000	8.000
0	30	5.000	.110	4.000	6.000
10	50	2.000	.028	1.500	2.500
10	40	10.000	.991	7.000	13.000
20	50	5.000	.110	4.000	6.000
30	60	3.000	.028	2.500	3.500
30	90	5.000	.110	4.000	6.000
40	70	10.000	.991	7.000	13.000
50	80	9.000	.991	6.000	12.000
60	80	3.001	.108	2.000	4.000
60	90	2.001	.027	1.500	2.500
70	100	5.001	.108	4.000	6.000
80	100	8.004	.970	5.000	11.000
90	100	4.001	.108	3.000	5.000

EVENT DISTRIBUTIONS

EVENT	FORWARD DISTRIBUTION MEAN	FORWARD DISTRIBUTION VARIANCE	BACKWARD DISTRIBUTION MEAN	BACKWARD DISTRIBUTION VARIANCE	SLACK
0	.000	.000	.000	.000	.000
10	3.000	.441	3.955	2.129	.955
30	5.000	.110	14.952	1.305	9.952
20	6.000	.441	6.954	2.346	.954
60	8.000	.140	17.951	1.101	9.951
90	10.253	.135	24.957	.108	14.704
50	11.000	.561	11.954	1.982	.954
40	13.000	1.441	13.957	1.104	.957
80	20.000	1.579	20.954	.970	.954
70	23.000	2.511	23.957	.108	.957
100	28.958	1.941	28.958	.000	.000

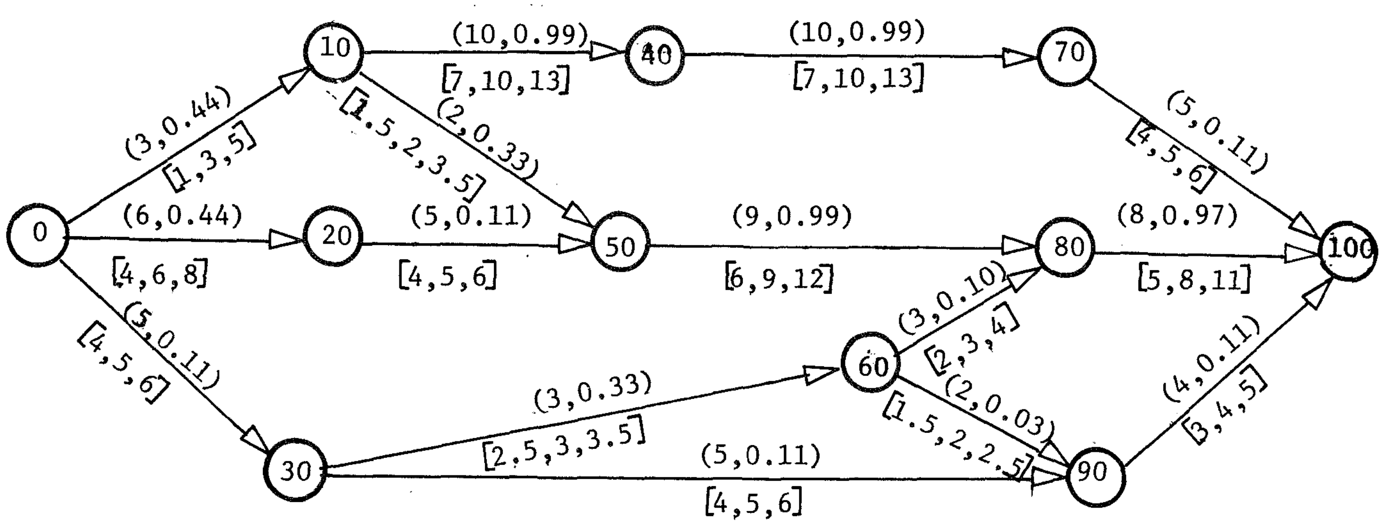


Figure 21. Network With Two Critical Paths

F_{100} is assumed to be approximately normal, we find that the probability of actually completing the project in the PERT-predicted time (28.0) is only about 0.25.

These examples demonstrate that it is not necessary to make the simplifying assumption of a critical path in performing the PERT calculations and that by using finite scanning techniques it is possible to accurately calculate event pdf's based on the activity pdf's that are input. Moreover, these examples have cast further doubts on the assumption that any single path need necessarily be more critical than any of the others. The procedures demonstrated here for calculating event times might readily be incorporated in a system for project management; one approach to designing this system will be explored in the following chapter

CHAPTER VI

SUMMARY

In this thesis we have examined the typical PERT implementation and the errors which can result from some of the standard PERT calculations. We have demonstrated a model for project management in which these calculations are performed correctly. In this chapter we will discuss the incorporation of a model such as the one introduced here in an information system for project management.

The basic input to any PERT system consists of activity information - for each activity, the bracketing events and distribution of durations would be input. The distribution could be enumerated, as in the case of our model, or, if the user felt that some standard form of pdf best represented activity duration, the distributions he desired could be generated by the system from appropriate input parameters (such as the mean and variance). In order to associate actual dates with events rather than merely times, it would be necessary to input the project starting date and to provide a calendar and the working schedule (number of working days in the week, holidays, etc.). This information could be input with the particular project, or it might be maintained internal to the system.

As a first step in the project initiation, given the activity information, the system would calculate the forward distribution of all events and would present f_∞ in graphic form on a CRT console, on-line plotter, or whatever device was available. As we discussed previously, the presentation of the entire f_∞, rather than merely μ_∞, provides the project manager more information on which to base his specification of t_∞. After he has chosen t_∞ on the basis of whatever criteria he has used (μ_∞, 95% confidence level, etc.), the system would perform the backward pass calculating the backward and slack distributions of all events.

With this project initiation run, the initial schedule would be produced and the network retained in storage. Typical output would include a list of all events along with information about their forward and backward distributions. The event information presented by our model, consisting of the means and variances of the forward and backward distributions along with the slack, might be representative of a minimum set of output; in addition the times associated with 95% confidence levels could be printed, the probability of occurrence of an event by some input target date, or even plots of the entire pdf's could be output. Another useful output would be an illustration of the PERT network, showing important time (e.g., Figure 9). This illustration would serve as a visual display of the project flow, and also it could be compared with a manually prepared original to ascertain that the network had been input properly.

The form in which the network is stored within the system would be dependent upon a tradeoff between time and storage requirements. For most applications it would probably be more economical to store only the input information and re-calculate the event distributions whenever a user desired to update or query the file. However, in a large sophisticated multiprogramming system it might be preferable to store more of the information about event distributions and to provide a query module to retrieve this information. The more powerful calculating routines would be called upon only as needed.

With the passing of time the system would monitor actual progress by comparing it with the predictions. Whenever any event occurred, or failed to occur on schedule, all event distributions, as well as the new probability of achieving t_∞, would be recalculated, and whenever desired (periodically, or on an exception basis) an updated schedule would be produced. As critical times approached, warnings could be produced along with requests for updated information, and whenever any event became overdue, a report of the fact along with its consequences would be generated (since event times are now represented by distributions rather than fixed times, it would be necessary for the user to specify the meaning of "overdue").

The system could include a "planning" mode of operation which would accept trial inputs, perform the necessary calculations to build a **temporary** network structure and respond to queries without altering the permanent project record. Using this mode the manager could update the consequences of the occurrence of some event on a particular date. Or he may wish to consider shifting some resource from one activity to another, by estimating the effects of this shift on the activity distribution, he may observe the effects on the overall project (increase or decrease of slack at various points, changes in probability of meeting t_∞, etc.) Given this planning tool, he can examine the consequence of a large variety of proposed actions.

With our model the concept of slack is no longer the simple difference between two fixed points in time as it is normally treated in PERT analysis. There is for any event i a distribution f_i representing the time at which the event is expected to occur and another distribution g_i representing the time at which it must occur in order that the project be completed by t_∞. There is then for any time t a probability that event i occurs t units of time ahead of its required time. We will call the function associated with this probability the <u>slack distribution</u>, denoted s_i. The probability that the slack of event i is exactly t is given by

$$pr(s_i = t) = pr(g_i(t_g)) \text{ and } pr(f_i(t_f)) \text{ for all } t = t_g - t_f, \text{ or}$$

$$s_i(t) = \int f_i(t_f) \cdot g_i(t_f + t)\, dt_f.$$

We note that this expression is equivalent to perform the serial composition, and the results derived previously, we see that $\mu(s_i) = \mu(g_i) - \mu(f_i)$. This value, $\mu(s_i)$, is that given the model as slack, and if a single parameter is to be used to represent the slack, this is probably the most meaningful. On the other hand, if the manager felt that additional information would be useful, the project management system would be capable of presenting the entire slack distribution function, in the same manner that it presents the forward and backward distribution.

This description of a project management information system indicates one possible means of utilizing the model described in Chapter IV. The system described is not inherently dependent upon our model, and there are PERT systems in existence which include many of the features which we have described. However, these systems are based upon critical path analysis, and we have seen that there are networks for which the critical path calculations introduce errors. The model which we have introduced can be utilized to calculate event distributions which accurately reflect the times predicted to perform the activities, no matter what the form of the network.

REFERENCES

1. Department of Defense Directive 3200.9, Institution of Engineering and Operational Systems Development, July 1, 1965.

2. MacCrimmon, R. R. and R. A. Ryavec: "An Analytic Study of the PERT Assumptions", Operations Research, 12:16-17 (1964).

3. Ringer, L. J.: Simulation of PERT Completion Times by Stratified Sampling Methods, Defense Documentation Center, AD 637-830, 1966.

4. Busacker, R. G. and T. L. Saaty: Finite Graphs and Networks, McGraw-Hill, New York, 1965.

5. Knuth, D. E.: The Art of Computer Programming, Vol. 1, "Fundamental Algorithms", Addison-Wesley, Reading, Mass., 1968.

6. Dixon, W.J. and F.J. Massey: Introduction to Statistical Analysis, McGraw-Hill, New York, 1957.

7. Lieberman, R.N.: Relational Structure Vertex Processor, University of Maryland Technical Report 69-87, March, 1969.

8. Hildebrand, F.B.: Advanced Calculus for Applications, Prentice Hall, Englewood Cliffs, N.J., 1962.

APPENDIX A

PROGRAM LISTING

```
@FOR,IS  MAIN
FOR S09-06/08-05:52 (,0)

   MAIN PROGRAM

   STORAGE USED: CODE(1) 000660; DATA(0) 000162; BLANK COMMON(2) 023423

    COMMON BLOCKS:

    0003   TLS     000001

   EXTERNAL REFERENCES (BLOCK, NAME)

    0004   SPACE
    0005   INITAL
    0006   CREATE
    0007   STODAT
    0010   INITIM
    0011   BUILD
    0012   STYPE
    0013   INDEX
    0014   MAVAR
    0015   STPFWD
    0016   STPBKD
    0017   NINTR$
    0020   NRDU$
    0021   NIO2$
    0022   NWDU$
    0023   NIO1$
    0024   SQRT
    0025   EXP
    0026   NSTOP$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001   000011 112G     0001   000012 115G     0001   000101 15L      0001   000167 175G     0001   000152 20L
    0001   000207 205G     0001   000225 215G     0001   000325 22L      0001   000275 227G     0001   000327 23L
    0001   000307 237G     0001   000403 27L      0001   000413 271G     0001   000426 300G     0001   000442 306G
    0001   000454 31L      0001   000504 322G     0001   000514 330G     0001   000521 335G     0001   000542 346G
    0001   000545 35L      0001   000572 367G     0001   000631 40L      0001   000641 405G     0001   000633 42L
    0000   000036 501F     0000   000065 611F     0000   000074 612F     0000   000107 613F     0000   000120 614F
    0000   000040 615F     0000   000047 616F     0000   000054 617F     0000 R 000035 CRIT     0000 R 000033 DELTIM
    0002 R 001750 EVNLST   0000 R 000024 FLOATI   0000 R 000023 H        0000 I 000013 I        0002   023421 IDATA
    0000 I 000017 IEV      0000 I 000015 IFRST    0000 I 000030 II       0013 I 000000 INDEX    0000 I 000022 INDX
    0002 I 005671 INTDIS   0003 I 000000 ITLS     0002 I 005670 ITOP     0000 I 000026 IUPPER   0000 I 000014 J
    0000 I 000021 JATOM    0002 I 023422 JDATA    0000 I 000027 JTOP     0000 I 000032 K        0000 I 000031 L
    0000 I 000016 LAST     0000 I 000025 LOWER    0002 I 000000 LSTEVN   0000 I 000020 NEXT     0000 R 000034 SD
    0004 R 000000 SPACE    0000 R 000000 TEMP     0002 R 005671 TIMDIS


00100      1*    C
00100      2*    C     MAIN EXECUTIVE ROUTINE. A LIST OF ALL EVENTS WITH ATOM NUMBERS IS
```

```
00100   C      KEPT IN LSTEVN. TIME DISTRIBUTIONS OF EVENTS AND ACTIVITIES ARE
00100   C      IN TIMDIS.
00100   C
00101          DIMENSION TEMP(11)
00103          EXTERNAL SPACE
00104          COMMON LSTEVN(500,2),EVNLST(500,4),ITOP,TIMDIS(7000),IDATA,JDATA
00105          DIMENSION INTDIS(1)
00106          COMMON /TLS/ ITLS
00107          EQUIVALENCE (INTDIS(1),TIMDIS(1))
00107   C
00107   C      INITIALIZE
00107   C
00110          CALL INITAL(SPACE,2000,0)
00111          DO 5 I=1,500
00114          DO 4 J=1,4
00117          EVNLST(I,J)=0.
00120        4 CONTINUE
00122        5 CONTINUE
00122   C
00122   C      READ FIRST AND LAST EVENTS, CREATE THEIR ATOMS.
00122   C
00124          READ (5,501) IFRST,LAST
00130      501 FORMAT (2(I3,1X))
00131          CALL CREATE(IEV,2)
00132          LSTEVN(1,1)=IFRST
00133          LSTEVN(1,2)=IEV
00134          CALL STODAT(IEV,TIMDIS(1))
00135          CALL INITIM(TIMDIS(2))
00136          INTDIS(1)=1
00137          CALL CREATE(IEV,2)
00140          LSTEVN(2,1)=LAST
00141          LSTEVN(2,2)=IEV
00142          CALL INITIM(TIMDIS(16))
00143          CALL STODAT(IEV,TIMDIS(15))
00144          INTDIS(15)=2
00144   C
00144   C      A  LIST OF EVENTS WHICH ARE READY FOR DISTRIBUTION CALCULATION
00144   C      IS MAINTAINED. ITLS POINTS TO THE TOP OF THIS LIST.
00144   C
00145          ITLS=1
00146          JDATA=30
00147          ITOP=2
00147   C
00147   C      BUILD THE PERT NETWORK.
00147   C
00150          CALL BUILD
00150   C
00150   C      STEP FORWARD ONE STEP. THE MEAN AND VARIATION OF THE NEXT EVENT
00150   C      IN THE TLS LIST ARE CALCULATED AND ITS CONTRIBUTION TO THE
00150   C      DISTRIBUTIONS OF SUCCEEDING EVENTS IS DETERMINED.
00150   C
00151       15 CONTINUE
00152          IF (ITLS.GE.9999) GO TO 20
00154          NEXT=ITLS
00155          JATOM=LSTEVN(NEXT,2)
00156          CALL STYPE(JATOM,3)
00157          INDX=INDEX(JATOM)
```

```
      ITLS=INTDIS(INDX-1)
      INTDIS(INDX-1)=NEXT
      CALL MAVAR(TIMDIS(INDX),EVNLST(NEXT,1),EVNLST(NEXT,2))
      CALL STPFWD(JATOM)
      IF (NEXT.NE.2) GO TO 15
   20 CONTINUE
C
C     WRITE DISTRIBUTION OF TERMINAL EVENT
C
      WRITE (6,615)
  615 FORMAT (1H1,35X,28H TERMINAL EVENT DISTRIBUTION)
      TEMP(1)=TIMDIS(INDX)
      H=(TIMDIS(INDX+1)-TIMDIS(INDX))/10.
      DO 17 I=1,10
      FLOATI=I
      TEMP(I+1)=TEMP(1)+FLOATI*H
   17 CONTINUE
      WRITE(6,616) TEMP
  616 FORMAT (1H0,5H TIME,18X,11(2X,F7.3))
      LOWER=INDX+2
      IUPPER=INDX+12
      DO 18 I=LOWER,IUPPER
      IF (TIMDIS(I).GT.0.0.AND.TIMDIS(I).LT.0.001) TIMDIS(I)=0.001
      IF (TIMDIS(I).GT.0.999.AND.TIMDIS(I).LT.1.0) TIMDIS(I)=0.999
   18 CONTINUE
      WRITE (6,617) (TIMDIS(I),I=LOWER,IUPPER)
  617 FORMAT (1X,23H CUMULATIVE PROBABILITY,10(5X,F4.3),4X,F5.3)
C
C     THE FORWARD DISTRIBUTIONS HAVE BEEN CALCULATED. THE NETWORK IS
C     STEPPED THROUGH FROM FINISH TO BEGINNING.
C
      EVNLST(2,3)=EVNLST(2,1)
      EVNLST(2,4)=EVNLST(2,2)
      DO 21 I=1,7000,14
      IF (INTDIS(I).NE.0) CALL INITIM(TIMDIS(I+1))
      IF (TIMDIS(I+10).LE.0.0) GO TO 22
   21 CONTINUE
   22 CONTINUE
      ITLS=2
   23 CONTINUE
      IF (ITLS.GE.9999) GO TO 27
      NEXT=ITLS
      JATOM=LSTEVN(NEXT,2)
      CALL STYPE(JATOM,4)
      INDX=INDEX(JATOM)
      ITLS=INTDIS(INDX-1)
      INTDIS(INDX-1)=NEXT
      CALL MAVAR(TIMDIS(INDX),EVNLST(NEXT,3),EVNLST(NEXT,4))
      EVNLST(NEXT,3)=EVNLST(2,1)-EVNLST(NEXT,3)
      CALL STPBKD(JATOM)
      IF (NEXT.NE.1) GO TO 23
   27 CONTINUE
C
C     SORT THE EVENTS BY MEAN TIME IN FORWARD DISTRIBUTION.
C
      DO 28 I=1,4
      EVNLST(1,I)=0.
```

```
00274      28 CONTINUE
00276         JTOP=ITOP+1
00277         DO 35 I=2,ITOP
00302         IF (EVNLST(I,1).GE.EVNLST(I-1,1)) GO TO 35
00304         II=I-1
00305         DO 30 J=II,2,-1
00310         IF (EVNLST(J,1).LT.EVNLST(I,1)) GO TO 31
00312      30 CONTINUE
00314         J=1
00315      31 CONTINUE
00316         J=J+1
00317         LSTEVN(JTOP,1)=LSTEVN(I,1)
00320         LSTEVN(JTOP,2)=LSTEVN(I,2)
00321         DO 32 L=1,4
00324         EVNLST(JTOP,L)=EVNLST(I,L)
00325      32 CONTINUE
00327         DO 33 K=II,J,-1
00332         LSTEVN(K+1,1)=LSTEVN(K,1)
00333         LSTEVN(K+1,2)=LSTEVN(K,2)
00334         DO 33 L=1,4
00337         EVNLST(K+1,L)=EVNLST(K,L)
00340      33 CONTINUE
00343         LSTEVN(J,1)=LSTEVN(JTOP,1)
00344         LSTEVN(J,2)=LSTEVN(JTOP,2)
00345         DO 34 L=1,4
00350         EVNLST(J,L)=EVNLST(JTOP,L)
00351      34 CONTINUE
00353      35 CONTINUE
00353   C
00353   C     WRITE ALL EVENTS.
00353   C
00355         WRITE (6,611)
00357     611 FORMAT(1H1,14X,22H EVENT DISTRIBUTIONS     //)
00360         WRITE (6,612)
00362     612 FORMAT  (60H EVENT  FORWARD DISTRIBUTION  BACKWARD DISTRIBUTION
00362        1SLACK              )
00363         WRITE (6,613)
00365     613 FORMAT(8X,40H MEAN    VARIANCE       MEAN   VARIANCE      /)
00366         DO 60 I=1,ITOP
00371         DELTIM=EVNLST(I,3)-EVNLST(I,1)
00372         IF (DELTIM.LT.0.001) GO TO 40
00374         SD=(SQRT(EVNLST(I,2))+SQRT(EVNLST(I,4)))/2.
00375         CRIT=1.0-EXP(-SD/(2.0*DELTIM))
00376         GO TO 42
00377      40 CONTINUE
00400         CRIT=1.0
00401      42 CONTINUE
00402         WRITE (6,614) LSTEVN(I,1),(EVNLST(I,J),J=1,4),DELTIM
00412     614 FORMAT (2X,I3,2X,F7.3,3X,F7.3,5X,F7.3,2X,F7.3,6X,F8.3      )
00413      60 CONTINUE
00415         STOP
00416         END

END OF COMPILATION:       NO  DIAGNOSTICS.
```

```
@FOR,IS  BUILD
FOR S09-06/08-05:52 (,0)


    SUBROUTINE BUILD      ENTRY POINT 000331


    STORAGE USED: CODE(1) 000337; DATA(0) 000110; BLANK COMMON(2) 023423


    EXTERNAL REFERENCES (BLOCK, NAME)

     0003    MAVAR
     0004    CREATE
     0005    STODAT
     0006    INITIM
     0007    POINTA
     0010    NWDU$
     0011    NI02$
     0012    NRDU$
     0013    NI01$
     0014    NERR3$


    STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

     0001    000143 12L       0001    000036 125G      0001    000154 14L       0001    000077 143G      0001    000131 160G
     0001    000172 17L       0001    000236 19L       0001    000305 25L       0001    000017 5L        0000    000046 501F
     0000    000012 601F      0000    000021 602F      0000    000035 603F      0000    000054 604F      0000 R 000006 ACTIV
     0000 R 000004 AMEAN      0002 R 001750 EVNLST     0000 I 000011 I          0002    023421 IDATA     0000 I 000001 IEVENT
     0000 I 000007 INIT       0000    000075 INJP$     0002 I 005671 INTDIS     0002 I 005670 ITOP      0002 I 023422 JDATA
     0000 I 000002 JEVENT     0000 I 000010 JTERM      0000 I 000000 KDATA      0002 I 000000 LSTEVN    0000 I 000003 M
     0002 R 005671 TIMDIS     0000 R 000005 VAR
```

```
00101             SUBROUTINE BUILD
00101        C
00101        C    SUBROUTINE READS IN THE ACTIVITY DISTRIBUTIONS AND BUILDS THE
00101        C    RELATIONSHIPS BETWEEN THE ACTIVITIES AND THE EVENTS. EACH CARD
00101        C    READ REPRESENTS ONE ACTIVITY. THE INFORMATION ON A CARD IS AS
00101        C    FOLLOWS...
00101        C      1. INITIATING EVENT
00101        C      2. TERMINATING EVENT
00101        C      3. TIME DISTRIBUTION FOR THE ACTIVITY
00101        C
00101        C    LSTEVN IS A STACK LISTING EACH EVENT WHICH HAS BEEN ENCOUNTERED
00101        C    WITH ITS ATOM NUMBER. ITOP POINTS TO THE TOP OF THE STACK.
00101        C
00103             COMMON LSTEVN(500,2),EVNLST(500,4),ITOP,TIMDIS(7000),IDATA,JDATA
00104             DIMENSION INTDIS(1)
00105             EQUIVALENCE (INTDIS,TIMDIS)
00105        C
00105        C    WRITE HEADERS.
00105        C
00106             WRITE (6,601)
```

```
00110        601 FORMAT (1H1,43X,24H ACTIVITY DISTRIBUTIONS      //)
00111            WRITE (6,602)
00113        602 FORMAT (39H INITIATING  TERMINATING       DURATION ,17X,
00113           1     20H INPUT DISTRIBUTION    )
00114            WRITE (6,603)
00116        603 FORMAT (44H   EVENT        EVENT        MEAN  VARIANCE        /)
00116      C
00116      C     READ ONE CARD AND CREATE AN ACTIVITY ATOM. THE INPUT IS TERMINATED
00116      C     BY A CARD WITH INITIATING EVENT EQUAL TO TERMINATING EVENT.
00116      C
00117          5 CONTINUE
00120            KDATA=JDATA+12
00121            READ(5,501) IEVENT,JEVENT,(TIMDIS(M),M=JDATA,KDATA)
00131        501 FORMAT (2(I3,1X),2(F7.3,1X),11(F4.3,1X))
00132            IF (IEVENT.EQ.JEVENT) RETURN
00134            CALL MAVAR (TIMDIS(JDATA),AMEAN,VAR)
00135            WRITE (6,604) IEVENT,JEVENT,AMEAN,VAR,(TIMDIS(M),M=JDATA,KDATA)
00147        604 FORMAT (4X,I3,9X,I3,6X,2(1X,F7.3),6X,2(1X,F7.3),10(1X,F4.3),1X,
00147           1     F5.3)
00150            CALL CREATE(ACTIV,1)
00151            CALL STODAT(ACTIV,TIMDIS(JDATA-1))
00152            JDATA=JDATA+14
00152      C
00152      C     DETERMINE THE ATOM NUMBERS OF THE INITIATING AND TERMINATING
00152      C     EVENTS.
00152      C
00153            INIT=0
00154            JTERM=0
00155            IF (ITOP.EQ.0) GO TO 17
00157            DO 15 I=1,ITOP
00162            IF (IEVENT.NE.LSTEVN(I,1)) GO TO 12
00164            INIT=LSTEVN(I,2)
00165            EVNLST(I,3)=EVNLST(I,3)+1.
00166         12 CONTINUE
00167            IF (JEVENT.NE.LSTEVN(I,1)) GO TO 14
00171            JTERM=LSTEVN(I,2)
00172            EVNLST(I,1)=EVNLST(I,1)+1.
00173         14 CONTINUE
00174            IF (INIT.GT.0.AND.JTERM.GT.0) GO TO 25
00176         15 CONTINUE
00200         17 CONTINUE
00201            IF (INIT.GT.0) GO TO 19
00201      C
00201      C     INITIATING EVENT IS NEW. CREATE A NEW ATOM.
00201      C
00203            ITOP=ITOP+1
00204            LSTEVN(ITOP,1)=IEVENT
00205            CALL CREATE(INIT,2)
00206            LSTEVN(ITOP,2)=INIT
00207            CALL STODAT(INIT,TIMDIS(JDATA-1))
00210            CALL INITIM(TIMDIS(JDATA))
00211            INTDIS(JDATA-1)=ITOP
00212            JDATA=JDATA+14
00213         19 CONTINUE
00214            IF (JTERM.GT.0) GO TO 25
00214      C
00214      C     TERMINATING EVENT IS NEW. CREATE A NEW ATOM.
```

```
00214   C
00216         ITOP=ITOP+1
00217         LSTEVN(ITOP,1)=JEVENT
00220         CALL CREATE(JTERM,2)
00221         LSTEVN(ITOP,2)=JTERM
00222         INTDIS(JDATA-1)=ITOP
00223         EVNLST(ITOP,1)=1.
00224         CALL STODAT(JTERM,TIMDIS(JDATA-1))
00225         CALL INITIM(TIMDIS(JDATA))
00226         JDATA=JDATA+14
00227      25 CONTINUE
00227   C
00227   C
00227   C     ADD TO INCOUNT ON TERMINATING ATOM
00227   C
00230         CALL POINTA(INIT,ACTIV)
00231         CALL POINTA(ACTIV,JTERM)
00232         GO TO 5
00233         END

     END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  STPEWD
FOR S09-06/08-05:52 (,0)

   SUBROUTINE STPFWD    ENTRY POINT 000142

   STORAGE USED: CODE(1) 000152; DATA(0) 000043; BLANK COMMON(2) 023423

    COMMON BLOCKS:

    0003   TLS    000001

   EXTERNAL REFERENCES (BLOCK, NAME)

    0004   JUMLOC
    0005   INDEX
    0006   SERIES
    0007   PARALL
    0010   ERROR
    0011   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001     000075 135G       0001    000104 26L        0001    000003 5L         0001    000130 50L        0001    000124 70L
    0002 R 001750 EVNLST     0000 I 000030 I          0002    023421 IDATA      0005 I 000000 INDEX      0000 I 000021 INDX
    0000     000033 INJP$      0002 I 005671 INTDIS     0003 I 000000 ITLS       0002 I 005670 ITOP       0000 I 000015 JATOM
    0002     023422 JDATA      0000 I 000020 JFLAG      0000 I 000017 JHIST      0000 I 000022 JNDX       0000 I 000016 JNUM
    0000 I 000023 KATOM      0000 I 000026 KFLAG      0000 I 000025 KHIST      0000 I 000027 KNDX       0000 I 000024 KNUM
    0002 I 000000 LSTEVN     0000 R 000000 TEMDIS     0002 R 005671 TIMDIS


00101      1*          SUBROUTINE STPFWD(INIEVT)
00101      2*    C
00101      3*    C     SUBROUTINE STEPS FORWARD ONE STEP FROM IVIEVT.
00101      4*    C
00103      5*          COMMON LSTEVN(500,2),EVNLST(500,4),ITOP,TIMDIS(7000),IDATA,JDATA
00104      6*          COMMON /TLS/ ITLS
00105      7*          DIMENSION INTDIS(1)
00106      8*          EQUIVALENCE (INTDIS(1),TIMDIS(1))
00107      9*          DIMENSION TEMDIS(13)
00107     10*    C
00107     11*    C     FIND AN ACTIVITY
00107     12*    C
00110     13*          JATOM=INIEVT
00111     14*          JNUM=0
00112     15*        5 CONTINUE
00113     16*          CALL JUMLOC(JNUM,JHIST,JATOM,JFLAG)
00114     17*          IF (JFLAG) 70,10,50
00117     18*       10 CONTINUE
00120     19*          INDX=INDEX(INIEVT)
00121     20*          JNDX=INDEX(JATOM)
00122     21*          CALL SERIES(TIMDIS(INDX),TIMDIS(JNDX),TEMDIS)
```

```
00122     C
00122     C      FIND THE EVENT
00122     C
00123            KATOM=JATOM
00124            KNUM=0
00125            CALL JUMLOC(KNUM,KHIST,KATOM,KFLAG)
00126            IF (KFLAG) 70,20,70
00131         20 CONTINUE
00132            KNDX=INDEX(KATOM)
00133            CALL PARALL(TEMDIS,TIMDIS(KNDX),TIMDIS(KNDX))
00133     C
00133     C      INCREMENT IN COUNTER.
00133     C
00134            DO 25 I=1,ITOP
00137            IF (LSTEVN(I,2).EQ.KATOM) GO TO 26
00141         25 CONTINUE
00143         26 CONTINUE
00144            EVNLST(I,2)=EVNLST(I,2)+1.
00145            IF (EVNLST(I,2).LT.EVNLST(I,1)) GO TO 5
00147            I=INTDIS(KNDX-1)
00150            INTDIS(KNDX-1)=ITLS
00151            ITLS=I
00152            GO TO 5
00153         70 CONTINUE
00154            CALL ERROR(72,JATOM)
00155         50 CONTINUE
00156            RETURN
00157            END

      END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  STPBKD
FOR S09-06/08-05:52 (,0)

   SUBROUTINE STPBKD     ENTRY POINT 000125

   STORAGE USED: CODE(1) 000134; DATA(0) 000043; BLANK COMMON(2) 023423

    COMMON BLOCKS:

    0003    TLS     000001

   EXTERNAL REFERENCES (BLOCK, NAME)

    0004    JUMLOP
    0005    INDEX
    0006    SERIES
    0007    PARALL
    0010    ERROR
    0011    NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001    000003 5L        0001    000114 50L       0001    000110 70L       0000 R 000031 ATOM       0002 R 001750 EVNLST
    0000 I 000030 I          0002    023421 IDATA     0005 I 000000 INDEX      0000 I 000021 INDX       0000   000034 INJP$
    0002 I 005671 INTDIS     0003 I 000000 ITLS      0002    005670 ITOP      0000 I 000015 JATOM      0002   023422 JDATA
    0000 I 000020 JFLAG      0000 I 000017 JHIST     0000 I 000022 JNDX       0000 I 000016 JNUM       0000 I 000023 KATOM
    0000 I 000026 KFLAG      0000 I 000025 KHIST     0000 I 000027 KNDX       0000 I 000024 KNUM       0002   000000 LSTEVN
    0000 R 000000 TEMDIS     0002 R 005671 TIMDIS


00101               SUBROUTINE STPBKD(INIEVT)
00101         C
00101         C     SUBROUTINE STEPS BACKWARD ONE STEP FROM EVENT INIEVT.
00101         C
00103               COMMON LSTEVN(500,2),EVNLST(500,4),ITOP,TIMDIS(7000),IDATA,JDATA
00104               COMMON /TLS/ ITLS
00105               DIMENSION INTDIS(1)
00106               EQUIVALENCE (INTDIS(1),TIMDIS(1))
00107               DIMENSION TEMDIS(13)
00107         C
00107         C     MOVE OUT NEXT ACTIVITY.
00107         C
00110               JATOM=INIEVT
00111               JNUM=0
00112             5 CONTINUE
00113               CALL JUMLOP(JNUM,JHIST,JATOM,JFLAG)
00114               IF (JFLAG) 70,10,50
00117            10 CONTINUE
00120               INDX=INDEX(INIEVT)
00121               JNDX=INDEX(JATOM)
00122               CALL SERIES(TIMDIS(INDX),TIMDIS(JNDX),TEMDIS)
```

```
00122       C
00122       C      MOVE TO PRECEDING EVENT.
00122       C
00123              KATOM=JATOM
00124              KNUM=0
00125              CALL JUMLOP(KNUM,KHIST,KATOM,KFLAG)
00126              IF (KFLAG) 70,20,70
00131           20 CONTINUE
00132              KNDX=INDEX(KATOM)
00133              CALL PARALL(TEMDIS,TIMDIS(KNDX),TIMDIS(KNDX))
00133       C
00133       C      INCREMENT OUT COUNTER.
00133       C
00134              I=INTDIS(KNDX-1)
00135              EVNLST(I,4)=EVNLST(I,4)+1.
00136              IF (EVNLST(I,4).LT.EVNLST(I,3)) GO TO 5
00140              INTDIS(KNDX-1)=ITLS
00141              ITLS=I
00142              GO TO 5
00143           70 CONTINUE
00144              CALL ERROR(72,ATOM)
00145           50 CONTINUE
00146              RETURN
00147              END

      END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  PARALL
FOR S09-06/08-05:52 (,0)

   SUBROUTINE PARALL    ENTRY POINT 000124

   STORAGE USED: CODE(1) 000144; DATA(0) 000043; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   CUMDIS
    0004   REFORM
    0005   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001    000013 110G      0001    000032 120G      0001    000063 133G      0001    000021 5L        0001    000040 8L
    0003 R 000000 CUMDIS    0000 R 000000 DISTRK    0000 R 000017 H         0000 I 000015 I         0000    000025 INJP$
    0000 I 000016 J         0000 I 000021 K         0000 R 000020 VALU


00101      1*          SUBROUTINE PARALL(DISTRI,DISTRJ,DISTRL)
00101      2*    C
00101      3*    C     THE COMPOSITE DISTRIBUTION REPRESENTED BY DISTRIBUTIONS DISTRI AND
00101      4*    C     DISTRJ IN PARALLEL IS COMPUTED AND PLACED IN DISTRK.
00101      5*    C
00103      6*          DIMENSION DISTRI(1),DISTRJ(1),DISTRL(1)
00104      7*          DIMENSION DISTRK(13)
00104      8*    C
00104      9*    C
00104     10*    C
00105     11*          IF (DISTRJ(2).GT.0.0) GO TO 5
00107     12*          DO 3 I=1,13
00112     13*        3 DISTRL(I)=DISTRI(I)
00114     14*          RETURN
00115     15*        5 IF (DISTRI(2).GT.0.0) GO TO 8
00117     16*          DO 7 J=1,13
00122     17*        7 DISTRL(J)=DISTRJ(J)
00124     18*          RETURN
00125     19*        8 CONTINUE
00125     20*    C
00125     21*    C     THE FIRST POINT IN DISTRK IS THE MAXIMUM OF THE FIRST POINTS IN
00125     22*    C     DISTRI AND DISTRJ. THE LAST POINT IN DISTRK IS MAXIMUM OF THE
00125     23*    C     LAST POINTS IN DISTRI AND DISTRJ.
00125     24*    C
00126     25*          DISTRK(1)=AMAX1(DISTRI(1),DISTRJ(1))
00127     26*          DISTRK(2)=AMAX1(DISTRI(2),DISTRJ(2))
00127     27*    C
00127     28*    C     DISTRK IS DIVIDED INTO 10 INTERVALS. VALU REPRESENTS A POINT ON
00127     29*    C     THE TIME AXIS OF DISTRK.
00127     30*    C
00130     31*          H=(DISTRK(2)-DISTRK(1))/10.
```

```
00131         VALU=DISTRK(1)
00132         DO 10 K=3,13
00132   C
00132   C     THE CUMULATIVE DISTRIBUTION AT A POINT IN DISTRK IS THE PRODUCT OF
00132   C     THE CUMULATIVE DISTRIBUTIONS OF DISTRI AND DISTRJ AT THAT POINT IN TIME.
00132   C
00135         DISTRK(K)=CUMDIS(DISTRI,VALU)*CUMDIS(DISTRJ,VALU)
00136         DISTRK(K)=ABS(DISTRK(K))
00137         VALU=VALU+H
00140      10 CONTINUE
00142         CALL REFORM(DISTRK,DISTRL)
00143         RETURN
00144         END

     END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  SERIES
FOR S09-06/08-05:52 (,0)

   SUBROUTINE SERIES    ENTRY POINT 000176

   STORAGE USED: CODE(1) 000221; DATA(0) 000061; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   CUMDIS
    0004   REFORM
    0005   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001     000027 110G      0001     000046 120G      0001     000104 135G      0001     000117 143G      0001     000035 5L
    0001     000147 50L       0001     000054 8L        0003 R 000000 CUMDIS    0000 R 000023 DISIVL    0000 R 000021 DISKVL
    0000 R 000000 DISTRK    0000 R 000017 HI        0000 R 000020 HK        0000 I 000015 I         0000     000027 INJP$
    0000 I 000016 J         0000 I 000022 K


00101               SUBROUTINE SERIES(DISTRI,DISTRJ,DISTRL)
00101          C
00101          C    THE COMPOSITE DISTRIBUTION REPRESENTED BY DISTRIBUTIONS DISTRI AND
00101          C    DISTRJ IN SERIES IS COMPUTED AND PLACED IN DISTRK.
00101          C
00101          C
00103               DIMENSION DISTRI(1),DISTRJ(1),DISTRL(1)
00104               DIMENSION DISTRK(13)
00104          C
00105               IF (DISTRJ(2).GT.0.0) GO TO 5
00107               DO 3 I=1,13
00112             3 DISTRL(I)=DISTRI(I)
00114               RETURN
00115             5 IF (DISTRI(2).GT.0.0) GO TO 8
00117               DO 7 J=1,13
00122             7 DISTRL(J)=DISTRJ(J)
00124               RETURN
00125             8 CONTINUE
00125          C
00125          C    THE FIRST POINT IN DISTRK IS THE SUM OF THE FIRST POINTS IN DISTRI
00125          C    AND DISTRJ. THE LAST POINT IN DISTRK IS THE SUM OF LAST POINTS IN
00125          C    DISTRI AND DISTRJ.
00125          C
00126               DISTRK(1)=DISTRI(1)+DISTRJ(1)
00127               DISTRK(2)=DISTRI(2)+DISTRJ(2)
00127          C
00127          C    HI IS THE STEP SIZE ALONG THE TIME AXIS IN DISTRI, HK IS THE STEP
00127          C    SIZE IN DISTRK. DISIVL IS A POINT ON THE TIME AXIS OF DISTRI,
00127          C    DISKVL A POINT ON THE TIME AXIS OF DISTRK. THE PROBABILITY OF
00127          C    FINISHING DISTRK IN THE MINIMUM TIME IS THE PRODUCT OF THE
```

```
00127   C      PROBABILITIES OF COMPLETING DISTRI AND DISTRJ IN THEIR MINIMUM TIMES.
00127   C
00130          HI=(DISTRI(2)-DISTRI(1))/10.
00131          HK=(DISTRK(2)-DISTRK(1))/10.
00132          DISTRK(3)=DISTRI(3)*DISTRJ(3)
00133          DISKVL=DISTRK(1)
00134          DO 50 K=4,13
00137          DISTRK(K)=0.
00140          DISKVL=DISKVL+HK
00141          DISIVL=DISTRI(1)+HI/2.
00141   C
00141   C      CALCULATE CUMULATIVE PROBABILITY FOR A POINT IN DISTRK.
00141   C
00142          DO 40 I=4,13
00145          IF (DISTRI(I).GT.DISKVL) GO TO 50
00147          DISTRK(K)=DISTRK(K)+(DISTRI(I)-DISTRI(I-1))*CUMDIS(DISTRJ,DISKVL
00147         1     -DISIVL)
00150          DISIVL=DISIVL+HI
00151       40 CONTINUE
00153       50 CONTINUE
00155          CALL REFORM(DISTRK,DISTRL)
00156          RETURN
00157          END

        END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  CUMDIS
FOR S09-06/08-05:52 (,0)

   FUNCTION CUMDIS      ENTRY POINT 000272

   STORAGE USED: CODE(1) 000311; DATA(0) 000062; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003    DEG3
    0004    NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001    000015 10L        0001    000030 120G       0001    000055 30L        0001    000252 50L        0000 R 000030 CF0
    0000 R 000031 CF1         0000 R 000032 CF2         0000 R 000033 CF3         0000 R 000020 COEF        0000 R 000030 COEFF
    0000 R 000024 COEFP       0000 R 000024 CP0         0000 R 000025 CP1         0000 R 000026 CP2         0000 R 000027 CP3
    0000 R 000000 CUMDIS      0000 R 000020 C0          0000 R 000021 C1          0000 R 000022 C2          0000 R 000023 C3
    0000 R 000001 H           0000 I 000004 I           0000   000040 INJP$       0000 I 000007 J           0000 I 000011 JM1
    0000 I 000013 JP1         0000 R 000015 P           0000 R 000017 PF          0000 R 000016 PP          0000 R 000006 PROBHI
    0000 R 000003 PROBLO      0000 R 000005 VALUHI      0000 R 000002 VALULO      0000 R 000010 X           0000 R 000014 XM1
    0000 R 000012 XP1


00101           FUNCTION CUMDIS(DISTR,VALU)
00101       C
00101       C   FUNCTION INTERPOLATES ON DISTRIBUTION DISTR TO FIND THE CUMULATIVE
00101       C   PROBABILITY FOR POINT VALU.
00101       C
00103           DIMENSION DISTR(1),COEF(4),COEFP(4),COEFF(4)
00104           EQUIVALENCE (C0,COEF(1)),(C1,COEF(2)),(C2,COEF(3)),(C3,COEF(4))
00105           EQUIVALENCE (CP0,COEFP(1)),(CP1,COEFP(2)),(CP2,COEFP(3)),
00105          1     (CP3,COEFP(4))
00106           EQUIVALENCE (CF0,COEFF(1)),(CF1,COEFF(2)),(CF2,COEFF(3)),
00106          1     (CF3,COEFF(4))
00106       C
00106       C   IF VALU IS LESS THAN THE LOW POINT ON DISTR, THE PROBABILITY IS 0.
00106       C
00107           IF (VALU.GE.DISTR (1)) GO TO 10
00111           CUMDIS=0.
00112           RETURN
00113        10 CONTINUE
00113       C
00113       C   H IS THE STEP SIZE ALONG THE TIME AXIS. FIND THE INTERVAL INTO
00113       C   WHICH VALU FALLS.
00113       C
00114           H=(DISTR(2)-DISTR(1))/10.
00115           VALULO=DISTR(1)
00116           PROBLO=DISTR(3)
00117           DO 20 I=4,13
00122           VALUHI=VALULO+H
```

```
00123              PROBHI=DISTR(I)
00124              IF (VALUHI.GT.VALU) GO TO 30
00126              PROBLO=PROBHI
00127              VALULO=VALUHI
00130           20 CONTINUE
00132              CUMDIS=1.
00133              RETURN
00134           30 CONTINUE
00135              I=I-1
00135        C
00135        C     J IS THE INDEX OF THE POINT AT THE LOW END OF THE INTERVAL INTO WHICH
00135        C     VALU FALLS. THREE THIRD DEGREE CURVES ARE DRAWN THROUGH THE POINTS
00135        C     IN THE VICINIY OF POINT J. X IS THE FRACTION OF AN INTERVAL BY
00135        C     WHICH VALU IS GREATER THAN TIME OF POINT J-1.
00135        C
00136              CALL DEG3(DISTR,I,COEF,J)
00137              X=ABS((VALU-DISTR(1)-H*FLOAT(J-3))/H)
00137        C
00137        C     CURVES ARE ALSO DETERMINED AROUND POINTS J-1 AND J+1.
00137        C
00140              CALL DEG3(DISTR,I-1,COEFP,JM1)
00141              XP1=ABS((VALU-DISTR(1)-H*FLOAT(JM1-3))/H)
00142              CALL DEG3(DISTR,I+1,COEFF,JP1)
00143              XM1=ABS((VALU-DISTR(1)-H*FLOAT(JP1-3))/H)
00143        C
00143        C     FIND CUMULATIVE DISTRIBUTION BASED ON THE FIRST CURVE.
00143        C
00144              P=C3*X**3+C2*X**2+C1*X+C0
00145              IF (X.LT.1.OR.X.GT.2) GO TO 50
00145        C
00145        C     FIND CUMULATIVE DISTRIBUTION LASE ON THE OTHER TWO CURVES AND
00145        C
00145        C     THE INTERVAL.
00145        C     PERFORM A WEIGHTER AVERAGE BASED ON THE POSITION OF VALU WITHIN
00147              PP=CP3*XP1**3+CP2*XP1**2+CP1*XP1+CP0
00150              PF=CF3*XM1**3+CF2*XM1**2+CF1*XM1+CF0
00151              P=(P+(2.-X)*PP+(X-1.)*PF)/2.
00152           50 CONTINUE
00153              CUMDIS=P
00154              RETURN
00155              END

       END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  DEG3
FOR S09-06/08-05:53 (,0)

   SUBROUTINE DEG3      ENTRY POINT 000104

   STORAGE USED: CODE(1) 000120; DATA(0) 000032; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0000   000015 INJP$     0000 R 000000 Y0      0000 R 000001 Y1      0000 R 000002 Y2      0000 R 000003 Y3


00101      1*          SUBROUTINE DEG3(ARRAY,I,COEF,J)
00101      2*    C
00101      3*    C     GENERATE COEFFICIENTS FOR A THIRD DEGREE POLYNOMIAL Y=P(X) PASSING
00101      4*    C     THROUGH FOUR POINTS OF ARRAY. AN ATTEMPT IS MADE TO FIT THE POINTS
00101      5*    C     IN THE FOLLOWING MANNER...
00101      6*    C        ARRAY(I-1)=Y(0)
00101      7*    C        ARRAY(I)  =Y(1)
00101      8*    C        ARRAY(I+1)=Y(2)
00101      9*    C        ARRAY(I+2)=Y(3).
00101     10*    C     IF THE POINTS UNDER CONSIDERATION ARE AT EITHER END OF ARRAY, AN
00101     11*    C     ADJUSTMENT MUST BE MADE. J IS SET TO THE INDEX OF THE POINT
00101     12*    C     ACTUALLY USED AS Y(0). THE COEFFICIENTS FOUND ARE RETURNED IN COEF.
00101     13*    C
00103     14*          DIMENSION ARRAY(1),COEF(1)
00104     15*          J=MIN0(MAX0(I-1,3),10)
00105     16*          Y0=ARRAY(J)
00106  *DIAGNOSTIC*  THE FOLLOWING STATEMENT IS REDUNDANT; HENCE, IT IS OMITTED.
00106     17*          Y1=ARRAY(J+1)
00107     18*          Y1=ARRAY(J+1)
00110     19*          Y2=ARRAY(J+2)
00111     20*          Y3=ARRAY(J+3)
00112     21*          COEF(1)=Y0
00113     22*          COEF(2)=(2.*Y3-9.*Y2+18.*Y1-11.*Y0)/6.
00114     23*          COEF(3)=(-Y3+4.*Y2-5.*Y1+2.*Y0)/2.
00115     24*          COEF(4)=(Y3-3.*Y2+3.*Y1-Y0)/6.
00116     25*          RETURN
00117     26*          END

       END OF COMPILATION:         1 DIAGNOSTICS.
```

```
@FOR,IS  REFORM
FOR S09-06/08-05:53 (,0)

   SUBROUTINE REFORM    ENTRY POINT 000124

   STORAGE USED: CODE(1) 000142; DATA(0) 000031; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   CUMDIS
    0004   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001   000062 10L      0001   000012 105G     0001   000074 136G      0001   000026 2L      0001   000042 5L
    0001   000046 7L       0003 R 000000 CUMDIS   0000 R 000001 HI        0000 R 000003 HJ      0000 I 000000 I
    0000   000014 INJP$    0000 I 000005 J        0000 R 000002 TIMEI     0000 R 000004 TIMEJ


00101      1*            SUBROUTINE REFORM(DISTRI,DISTRJ)
00101      2*    C
00101      3*    C       REFORMAT DISTRIBUTION DISTRI INTO DISTRJ. THE LIMITS ARE MODIFIED
00101      4*    C       TO INCLUDE ONLY POINTS WHOSE CUMULATIVE PROBABILITIES LIE BETWEEN
00101      5*    C       .005 AND .995.
00101      6*    C
00103      7*            DIMENSION DISTRI(1),DISTRJ(1)
00103      8*    C
00103      9*    C       FIND LOWER END OF RESULTANT DISTRIBUTION.
00103     10*    C
00104     11*            DO 1 I=1,13
00107     12*          1 DISTRJ(I)=DISTRI(I)
00111     13*            RETURN
00112  *DIAGNOSTIC*      CONTROL CAN NEVER REACH THE NEXT STATEMENT
00112     14*            HI=(DISTRI(2)-DISTRI(1))/100.
00113     15*            TIMEI=DISTRI(1)
00114     16*          2 CONTINUE
00115     17*            IF (CUMDIS(DISTRI,TIMEI).GE.0.0005) GO TO 5
00117     18*            TIMEI=TIMEI+HI
00120     19*            GO TO 2
00121     20*          5 CONTINUE
00122     21*            DISTRJ(1)=TIMEI
00122     22*    C
00122     23*    C       FIND UPPER END OF RESULTANT.
00122     24*    C
00123     25*            TIMEI=DISTRI(2)
00124     26*          7 CONTINUE
00125     27*            IF (CUMDIS(DISTRI,TIMEI).LE.0.9995) GO TO 10
00127     28*            TIMEI=TIMEI-HI
00130     29*            GO TO 7
00131     30*         10 CONTINUE
00131     31*    C       COMPUTE PROBABILITY POINTS OF DISTRIBUTION.
```

```
00131     C
00132              DISTRJ(2)=TIMEI
00133              HJ=(DISTRJ(2)-DISTRJ(1))/10.
00134              TIMEJ=DISTRJ(1)
00135              DO 20 J=3,13
00140              DISTRJ(J)=CUMDIS(DISTRI,TIMEJ)
00141              TIMEJ=TIMEJ+HJ
00142           20 CONTINUE
00144              DISTRJ(3)=0.
00145              DISTRJ(13)=1.0
00146              RETURN
00147              END

      END OF COMPILATION:          1 DIAGNOSTICS.
```

```
@FOR,IS  INITIM
FOR S09-06/08-05:53 (,0)

   SUBROUTINE INITIM    ENTRY POINT 000036

   STORAGE USED: CODE(1) 000046; DATA(0) 000020; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001   000012 107G      0000 I 000000 I         0000   000004 INJP$


00101          SUBROUTINE INITIM(DIST)
00101     C
00101     C    INITIALIZE THE TIME DISTRIBUTION OF AN EVENT.
00101     C
00103          DIMENSION DIST(1)
00104          DIST(1)=0.
00105          DIST(2)=0.
00106          DO 10 I=3,13
00111          DIST(I)=1.00-.000013+.000001*FLOAT(I)
00112       10 CONTINUE
00114          RETURN
00115          END

      END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  MAVAR
FOR S09-06/08-05:53 (,0)

   SUBROUTINE MAVAR     ENTRY POINT 000075

   STORAGE USED: CODE(1) 000114; DATA(0) 000043; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   PNTGEN
    0004   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001   000020 111G       0001   000032 116G       0000 R 000013 H        0000 I 000017 I        0000   000025 INJP$
    0000 I 000020 J          0000 R 000000 POINT      0000 R 000021 PROB     0000 R 000015 SUM      0000 R 000016 SUMSQ
    0000 R 000014 X


00101      1*          SUBROUTINE MAVAR(DIST,AMEAN,VAR)
00101      2*     C
00101      3*     C    CULCULATE THE MEAN AND VARIATION OF DISTRIBUTION DIST. THE MEAN
00101      4*     C    IS PLACED IN AMEAN, THE VARIATION IN VAR.
00101      5*     C
00103      6*          DIMENSION DIST(1),POINT(11)
00104      7*          H=(DIST(2)-DIST(1))/100.
00105      8*          X=DIST(1)+H/2.
00106      9*          SUM=0.
00107     10*          SUMSQ=0.
00110     11*          DO 30 I=1,10
00113     12*          CALL PNTGEN(DIST,I,POINT(2))
00114     13*          POINT(1)=DIST(I+2)
00115     14*          DO 20 J=1,10
00120     15*          PROB=POINT(J+1)-POINT(J)
00121     16*          SUM=SUM+PROB*X
00122     17*          SUMSQ=SUMSQ+PROB*X**2
00123     18*          X=X+H
00124     19*       20 CONTINUE
00126     20*       30 CONTINUE
00130     21*          AMEAN=SUM
00131     22*          VAR=SUMSQ-SUM**2
00132     23*          RETURN
00133     24*          END

     END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
QFOR,IS  PNTGEN
FOR S09-06/08-05:53 (,0)

   SUBROUTINE PNTGEN    ENTRY POINT 000266

   STORAGE USED: CODE(1) 000306; DATA(0) 000056; BLANK COMMON(2) 000000

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003    DEG3
    0004    NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001    000113 120G      0001    000217 5L       0000 R 000021 CF0      0000 R 000022 CF1      0000 R 000023 CF2
    0000 R 000024 CF3       0000 R 000011 COEF      0000 R 000021 COEFF    0000 R 000015 COEFP    0000 R 000015 CP0
    0000 R 000016 CP1       0000 R 000017 CP2       0000 R 000020 CP3      0000 R 000011 C0       0000 R 000012 C1
    0000 R 000013 C2        0000 R 000014 C3        0000   000031 INJP$    0000 I 000004 J        0000 I 000002 K
    0000 I 000003 KF        0000 I 000001 KP        0000 R 000010 PF       0000 R 000007 PP       0000 R 000000 X
    0000 R 000006 XM1       0000 R 000005 XP1
```

```
00101             SUBROUTINE PNTGEN (DIST,I,PNT)
00101        C    GENERATE 10 PROBABILITY POINTS BETWEEN POINT I AND POINT I+1 IN
00101        C    DISTRIBUTION DIST. THE POINTS ARE PLACED IN ARRAY PNT.
00101        C
00103             DIMENSION DIST(1),PNT(1),COEF(4),COEFP(4),COEFF(4)
00104             EQUIVALENCE (C0,COEF(1)),(C1,COEF(2)),(C2,COEF(3)),(C3,COEF(4))
00105             EQUIVALENCE (CP0,COEFP(1)),(CP1,COEFP(2)),(CP2,COEFP(3)),
00105            1     (CP3,COEFP(4))
00106             EQUIVALENCE (CF0,COEFF(1)),(CF1,COEFF(2)),(CF2,COEFF(3)),
00106            1     (CF3,COEFF(4))
00107             X=1.
00110             IF (I.LE.1) X=0.
00112             IF (I.GE.10) X=2.
00112        C
00112        C    GENERATE 3 POLYNOMIALS PASSING THROUGH POINT I.
00112        C
00114             CALL DEG3(DIST,I+1,COEFP,KP)
00115             CALL DEG3(DIST,I+2,COEF,K)
00116             CALL DEG3(DIST,I+3,COEFF,KF)
00117             DO 10 J=1,10
00122             X=X+.1
00122        C
00122        C    FOR EACH POINT J FIND THE CUMULATIVE PROBABILITY ON EACH OF THE
00122        C    THREE POLYNOMIALS.
00122        C
00123             PNT(J)=C3*X**3+C2*X**2+C1*X+C0
00124             IF (I.LE.1.OR.I.GE.10) GO TO 5
00126             XP1=X
00127             XM1=X
```

```
00130    30*          IF (I.LT.9) XM1=X-1.
00132    31*          IF (I.GT.2) XP1=X+1.
00134    32*          PP=CP3*XP1**3+CP2*XP1**2+CP1*XP1+CP0
00135    33*          PF=CF3*XM1**3+CF2*XM1**2+CF1*XM1+CF0
00135    34*    C
00135    35*    C     TAKE A WEIGHTED AVERAGE OF THE PROBABILITIES BASED UPON THE
00135    36*    C     DISTANCE OF POINT J ALONG THE INTERVAL BETWEEN POINTS I AND I+1.
00135    37*    C
00136    38*          PNT(J)=(PNT(J)+(2.-X)*PP+(X-1.)*PF)/2.
00137    39*        5 CONTINUE
00140    40*          PNT(J)=AMIN1(PNT(J),DIST(I+3))
00141    41*          PNT(1)=AMAX1(PNT(1),DIST(I+2))
00142    42*          IF (J.GT.1) PNT(J)=AMAX1(PNT(J),PNT(J-1))
00144    43*       10 CONTINUE
00146    44*          RETURN
00147    45*          END

       END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@FOR,IS  MABSTF
FOR S09-06/08-05:53 (,0)

   FUNCTION MAB          ENTRY POINT 000110
            MABDAT       ENTRY POINT 000113
            MABTYP       ENTRY POINT 000121
            INDEX        ENTRY POINT 000127

   STORAGE USED: CODE(1) 000135; DATA(0) 000014; BLANK COMMON(2) 023423

   EXTERNAL REFERENCES (BLOCK, NAME)

    0003   FIND
    0004   ERROR
    0005   NABDAT
    0006   NABTYP
    0007   NABADD
    0010   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0002    001750 EVNLST    0003 R 000000 FIND      0000 I 000001 IADRS     0002    023421 IDATA     0000    000006 INJP$
    0002    005670 ITOP      0002    023422 JDATA    0002    000000 LSTEVN   0000 I 000000 MAB       0007 I 000000 NABADD
    0005 I 000000 NABDAT     0006 I 000000 NABTYP    0002 R 005671 TIMDIS


00101      1*          FUNCTION MAB(X)
00101      2*    C
00101      3*    C     UTILITY FUNCTIONS MANIPULATING RSVP STRUCTURES.
00101      4*    C
00103      5*          COMMON LSTEVN(500,2),EVNLST(500,4),ITOP,TIMDIS(7000),IDATA,JDATA
00104      6*          ENTRY MABDAT(NUMB1)
00104      7*    C
00104      8*    C     RECOVER THE DATA POINTER OF ATOM NUMB1.
00104      9*    C
00106     10*          IF (FIND(NUMB1,IADRS,0).LT.0.)CALL ERROR(69,NUMB1)
00110     11*          MAB=NABDAT(IADRS)+1
00111     12*          RETURN
00112     13*          ENTRY MABTYP(NUMB2)
00112     14*    C     FIND THE TYPE OF ATOM NUMB2.
00112     15*    C
00112     16*    C
00114     17*          IF (FIND(NUMB2,IADRS,0).LT.0.) CALL ERROR (70,NUMB2)
00116     18*          MAB=NABTYP(IADRS)
00117     19*          RETURN
00120     20*          ENTRY INDEX(NUMB3)
00120     21*    C
00120     22*    C     FIND INDEX TO DATA OF ATOM NUMB3.
00120     23*    C
00122     24*          IF (FIND(NUMB3,IADRS,0).LT.0.) CALL ERROR(71,NUMB3)
00124     25*          MAB=NABDAT(IADRS)-NABADD(TIMDIS)+2
```

```
00125    26*         RETURN
00126    27*         END

      END OF COMPILATION:        NO  DIAGNOSTICS.
```