Application Survey Paper

Project Planning with PERT/CPM

Program Evaluation and Review Technique (PERT) and Critical Path Method (CPM) are two closely related techniques for monitoring the progress of a large project. A key part of PERT/CPM is calculating the critical path. That is, identifying the subset of the activities that must be performed exactly as planned in order for the project to finish on time.

PERT became famous when it was developed and used for the management of the development of the Polaris fleet ballistic missile system for the U.S. Navy. This project was notable in that it finished 18 months ahead of schedule and within budget. At roughly the same time, the DuPont company was using CPM to manage its construction and repair of manufacturing plants.

Crashing, Resource Constraints, and Uncertainty

A typical modern-day project has a variety of complications not considered in the original PERT/CPM methodology. We look at three particular situations: a) You may be able to speed the completion of a project by speeding up or “crashing” some of the activities in the project. b) Your ability to finish a project quickly is hindered by limited resources (e.g., two activities that might otherwise be done simultaneously, in fact have to be done sequentially because they both require a crane and you have only one crane on site). c) How long it takes to do each activity is a random variable. Therefore, how long it takes to do the entire project is a random variable. Given the uncertainty of each activity, what can we say about the probability that the entire project will finish by a given target date?

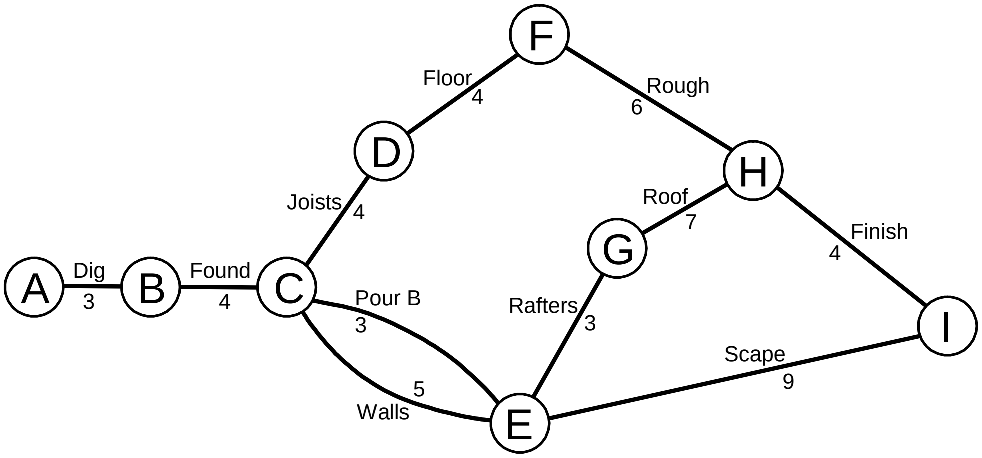
We will start by looking at a simple example of a standard PERT/CPM without complications. The calculation of the critical path is conceptually simple, although for large projects it convenient to automate it.

In the table below, we list the activities involved in the simple, but nontrivial, project of building a house. An activity cannot be started until all of its predecessors are finished:

Activity	Mnemonic	Activity Time	Predecessors (Mnemonic)
Dig Basement	DIG	3	—
Pour Foundation	FOUND	4	DIG
Pour Basement	POURB	3	FOUND
Install Floor Joists	JOISTS	4	FOUND
Install Walls	WALLS	5	FOUND
Install Rafters	RAFTERS	3	WALLS, POURB
Install Flooring	FLOOR	4	JOISTS
Rough Interior	ROUGH	6	FLOOR
Install Roof	ROOF	7	RAFTERS
Finish Interior	FINISH	4	ROUGH, ROOF
Landscape	SCAPE	9	POURB, WALLS

In Figure 1, we show the so-called PERT (or activity-on-arrow) network for this project. We would like to calculate the minimum elapsed time to complete this project. Relative to this figure, the number of interest is simply the longest path from left to right in this figure. The project can be completed no sooner than the sum of the times of the successive activities on this path. Clearly, *DIG* and *FOUND* must be on the critical path. Also, at least one of *FINISH* and *SCAPE* must be on the critical path. Verify for yourself that the critical path consists of activities *DIG****,*** *FOUND*, *WALLS*, *RAFTERS*, *ROOF*, and *FINISH* and has a length of 26.

Figure 1 Activity-on-Arc PERT/CPM Network



Activity-on-Arc vs. Activity-on-Node Network Diagrams

Two conventions are used in practice for displaying project networks: (1) Activity-on-Arc (AOA) and (2) Activity-on-Node (AON). Our previous example used the AOA convention. The characteristics of the two are:

AON

- Each activity is represented by a node in the network.
- A precedence relationship between two activities is represented by an arc or link between the two.
- AON may be less error prone because it does not need dummy activities or arcs.

AOA

- Each activity is represented by an arc in the network.
- If activity *X* must precede activity *Y*, there are *X* leads into arc *Y*. Thus, the nodes represent events or “milestones” (e.g., “finished activity *X*”). Dummy activities of zero length may be required to properly represent precedence relationships.
- AOA historically has been more popular, perhaps because of its similarity to Gantt charts used in scheduling.

A small project with six activities is displayed in *AON* form in Figure 2. The number next to each node is the duration of the activity. By inspection, you can discover that the longest path consists of activities *A*, *C*, *E*, and *F*. It has a length of 29. The corresponding *AOA* network for the same project is shown in Figure 3. In the *AOA* network, we have enclosed the activity letters in circles above the associated arc. The unenclosed numbers below each arc are the durations of the activities. We have given the nodes, or milestones, arbitrary number designations enclosed in squares. Notice the dummy activity (the dotted arc) between nodes 3 and 4. This is because *a dummy activity will be required in an AOA diagram anytime that two activities* (e.g., *A* and *B*) *share some* (e.g., activity *D*), *but not all* (e.g., activity *C*), *successor activities.*

Figure 2 An Activity-on-Node Representation

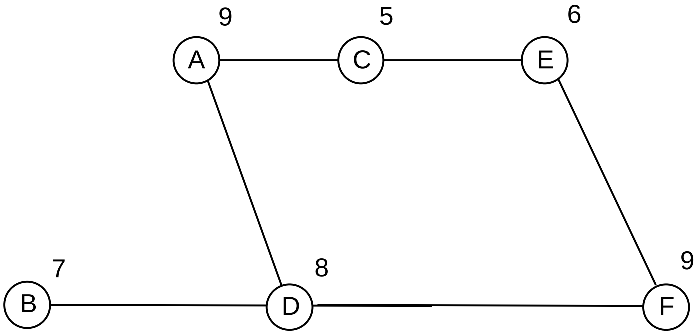
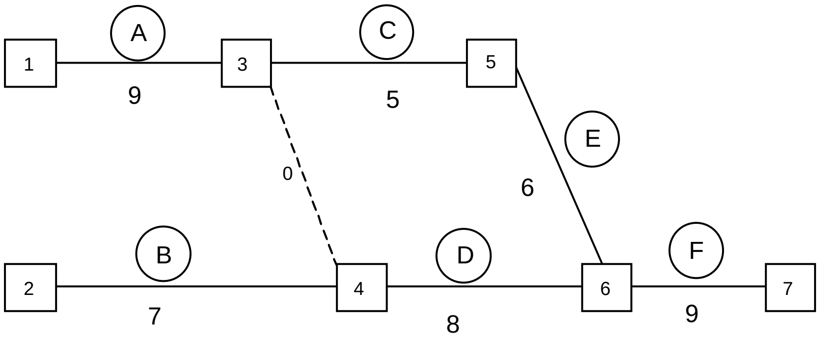


Figure 3 An Activity-on-Arc Representation



Crashing of Project Networks

Once the critical path length for a project has been identified, the next question invariably asked is: can we shorten the project? The process of decreasing the duration of a project or activity is commonly called crashing. For many construction projects, it is common for the customer to pay an incentive to the contractor for finishing the project in a shorter length of time. For example, in highway repair projects, it is not unusual to have incentives from $5,000 to $25,000 per day that the project is finished before a target date.

The Cost and Value of Crashing

There is value in crashing a project. In order to crash a project, we must crash one or more activities. Crashing an activity costs money. Deciding to crash an activity requires us to compare the cost of crashing that activity with the value of the resulting reduction in project length. This decision is frequently complicated by the fact that some negotiation may be required between the party that incurs the cost of crashing the activity (e.g., the contractor) and the party that enjoys the value of the crashed project (e.g., the customer).

The Cost of Crashing an Activity

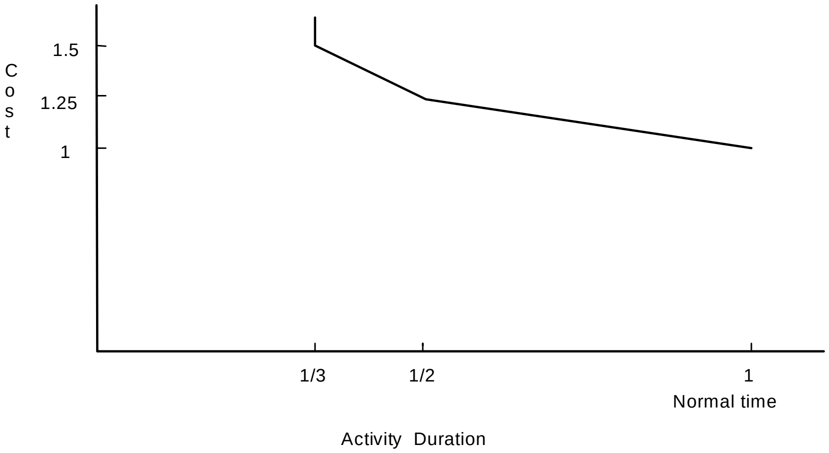
An activity is typically crashed by applying more labor to it (e g., overtime or a second shift). We might typically expect that using second-shift labor could cost 1.5 times as much per hour as first-shift labor. We might expect third-shift labor to cost twice as much as first-shift labor.

Consider an activity that can be done in six days if only first-shift labor is used and has a labor cost of $6,000. If we allow the use of second-shift labor and thus work two shifts per day, the activity can be done in three days for a cost of $3 \times 1000 + 3 \times l000 \times 1.5 = 7,500$. If third-shift labor is allowed, then the project can be done in two days by working three shifts per day and incurring a total of:

$$2 \times 1000 + 2 \times 1000 \times 1.5 + 2 \times 1000 \times 2 = \$9,000.$$

Thus, we get a crashing cost curve for the activity as shown in Figure 4:

Figure 4 Activity Crash Cost Curve



The Value of Crashing a Project

There are two approaches to deciding upon the amount of project crashing: (a) we simply specify a project duration time and crash enough to achieve this duration, or (b) we estimate the value of crashing it for various days. As an example of (a), in 1987 a new stadium was being built for the Montreal Expos baseball team. The obvious completion target was the first home game of the season.

As an example of (b), consider an urban expressway repair. What is the value per day of completing it early? Suppose that 6,000 motorists are affected by the repair project and each is delayed by 10 minutes each day because of the repair work (e.g., by taking alternate routes or by slower traffic). The total daily delay is 6,000 × 10 = 60,000 minutes = 1000 hours. If we assign an hourly cost of $5/person × hours, the social value of reducing the repair project by one day is $5,000.

Formulation of the Crashing Problem

Suppose we have investigated the crashing possibilities for each activity or task in our previous project example. These estimates are summarized in the following table:

Activity	Predecessor	Normal duration (Days)	Minimum duration if crashed (Days)	$/Day
A	—	9	5	5000
B	—	7	3	6000
C	*A*	5	3	4000
D	*A,B*	8	4	2000
E	*C*	6	3	3000
F	*D,E*	9	5	9000

For example, activity *A* could be done in five days rather than nine. However, this would cost us an extra (9 – 5) × 5000 = $20,000.

First, consider the simple case where we have a hard due date by which the project must be done. Let us say 22 days in this case. How would we decide which activities to crash? Activity *B* is the cheapest to crash per day. However, it is not on the critical path, so its low cost is at best just interesting.

Let us define:

EF_i = earliest finish time of activity *i*, taking into account any crashing that is done;
C_i = number of days by which activity *i* is crashed.

In words, the LP model will be:

Minimize Cost of crashing
subject to
For each activity *j* and each predecessor *i*:
earliest finish of *j* ≥ earliest finish of predecessor *i* + actual duration of *j*;
For each activity *j:*
minimum duration for *j* if crashed ≤ actual duration of *j* ≤ normal duration for *j*.

A LINGO formulation for this can be found in model PERTCRASH.

It turns out that for an additional cost of $31,000, we can meet the 22 day deadline.

Now, suppose there is no hard project due date, but we do receive an incentive payment of $5000 for each day we reduce the project length. Define *PCRASH* = number of days the project is finished before the twenty-ninth day. Now, the formulation in words is:

Maximize Incentive_payments_received – cost_of_crashing;

subject to

For each activity *j* and each predecessor *i*:
earliest finish of *j* ≥ earliest finish of predecessor *i* + actual duration of *j*;
For each activity *j*:
minimum duration for *j* if crashed ≤ actual duration of *j* ≤ normal duration for *j*.

From the solution, we see we should crash it by five days to give a total project length of twenty-four days, and increase our net revenues by $6000:

```
Objective value:   6000.000
            Variable           Value        Reduced Cost
              PCRASH        5.000000           0.0000000
              EF( A)        7.000000           0.0000000
              EF( B)        7.000000           0.0000000
              EF( C)        12.00000           0.0000000
              EF( D)        15.00000           0.0000000
              EF( E)        15.00000           0.0000000
              EF( F)        24.00000           0.0000000
          ACTUAL( A)        7.000000           0.0000000
          ACTUAL( B)        7.000000           -6000.000
          ACTUAL( C)        5.000000           -1000.000
          ACTUAL( D)        8.000000           0.0000000
          ACTUAL( E)        3.000000           0.0000000
          ACTUAL( F)        9.000000           -4000.000
```

The excess of the incentive payments over crash costs is $6,000.

Figure 5 captures the complete relationship between project length and the cost of crashing. Notice that the first few days of reduction in project length cost only $3000 per day. The last reductions, from 20 days to 16 days, costs $9000 per day.

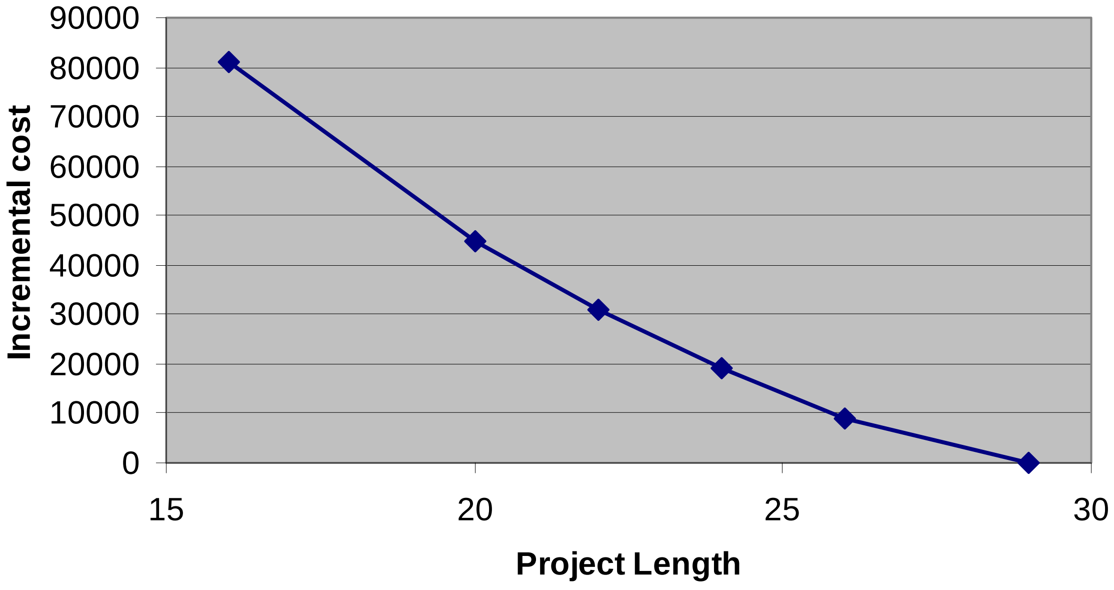


Figure 5 Project Cost as a Function of Project Length

We have assumed that an activity can be crashed continuously between a maximum length and a minimum length. An alternative situation sometimes encountered is that there are two alternative ways of doing a project, an expensive fast way, or a cheap slow way, with no “in between” way. We do not discuss that situation here. Nevertheless, it can be modeled by use of binary variables.

Resource Constraints in Project Scheduling

For many projects, a major complication is that there are a limited number of resources. The limited resources require you to do tasks sequentially that otherwise might be done simultaneously. When we have project with resource constraints, we have not only the standard precedence constraints, but also: a) a set of resources, each with a known capacity constant over time, and b) for each activity a list of how much it requires of each resource when the task is being performed. We cannot perform a set of tasks simultaneously if they together require more of some resource than is available.

We will illustrate with an extension of our house building project. There are two scarce resources, *Supervisor* time and *Carpenter* time

```
DATA:
! Task names and duration,
 duration times must be integer;
  TASK  TIME =
 DIG      3
 FOUND    4
 JOISTS   4
 POURB    3
 WALLS    5
```

```
  FLOOR    4
  RAFTERS  3
  ROUGH    6
  ROOF     7
  FINISH   4
  SCAPE    9 ;
!The precedence constraint pairs;
 PRED =
  DIG     FOUND
  FOUND   POURB
  FOUND   JOISTS
  FOUND   WALLS
  WALLS   RAFTERS
  POURB   RAFTERS
  JOISTS  FLOOR
  FLOOR   ROUGH
  RAFTERS ROOF
  ROUGH   FINISH
  ROOF   FINISH
  POURB   SCAPE
  WALLS   SCAPE;
! The scarce resources with their capacity;
   RESOURCE = SUPERV  CARPENTER;
        CAP =    1,        2;
! How much each task needs of each resource;
          TXR,      NEED =
    DIG     SUPERV    1
    FOUND   SUPERV   .5
    WALLS   SUPERV   .4
    POURB   SUPERV   .3
    JOISTS  SUPERV   .3
    FLOOR   SUPERV   .4
    ROUGH   SUPERV   .1
    RAFTERS SUPERV   .3
    ROOF    SUPERV   .2
    FINISH  SUPERV   .6
    SCAPE   SUPERV   .6
    FOUND   CARPENTER 2
    JOISTS  CARPENTER 2
    WALLS   CARPENTER 1
    FLOOR   CARPENTER 1
    ROUGH   CARPENTER 1
    RAFTERS CARPENTER 2
    ROOF    CARPENTER 1
    FINISH  CARPENTER 1;
 !  Upper  limit  on  number  of  periods  required  to  complete  the
project;
   PERIOD = 1..40;
 ENDDATA
```

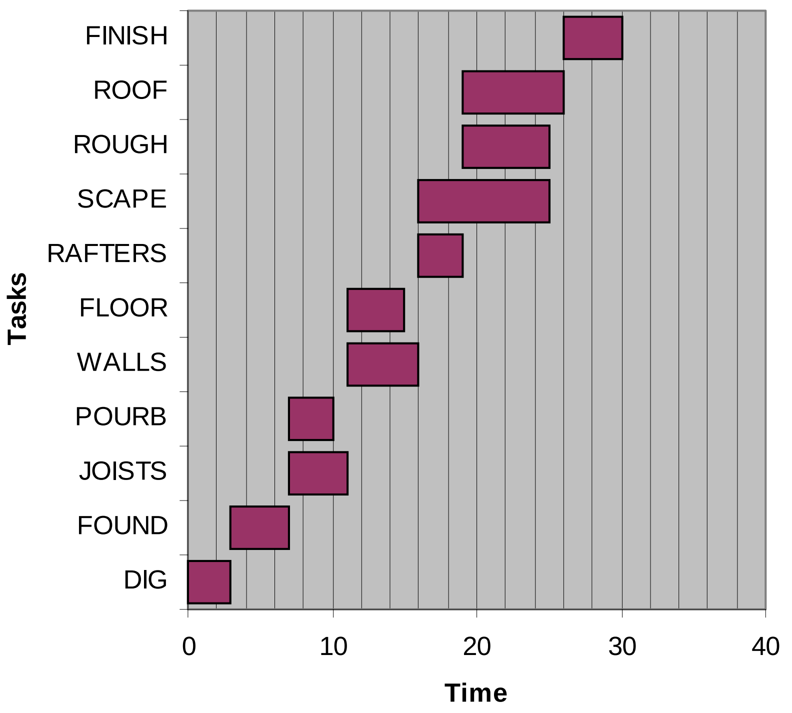
Notice from the data that *FINISH* and *SCAPE* cannot be done simultaneously. The total supervisor time required, .6 + .6 = 1.2, is greater than that available. Also, *FLOOR* and *RAFTERS* cannot be done simultaneously because the total carpenter time required, 1 + 2 = 3, is greater than the carpenter time available. Recall that with unlimited resources the project could be done in 26 days. Will conflicts such as just mentioned cause a substantial increase in project length?

We shall see that with careful scheduling, the project length need increase only to 30 days. The following start times will achieve a project length of 30 days.

Task	Start at beginning of day
DIG	1
FOUND	4
JOISTS	8
POURB	8
WALLS	12
FLOOR	12
RAFTERS	17
SCAPE	17
ROUGH	20
ROOF	20
FINISH	27

Figure 6 shows this schedule in Gantt chart form. You can verify that on no day are a) more than two carpenters required, or b) more than one supervisor required. Notice that of the two conflicting tasks, *SCAPE* and *FINISH*, *SCAPE* is started early enough, on day 17, so that it finishes by the time that *FINISH* is started at the beginning of day 27. The duration of 4 days for *FINISH* means that the project finishes at the end of day 30.

Figure 6 Gantt Chart for House Project



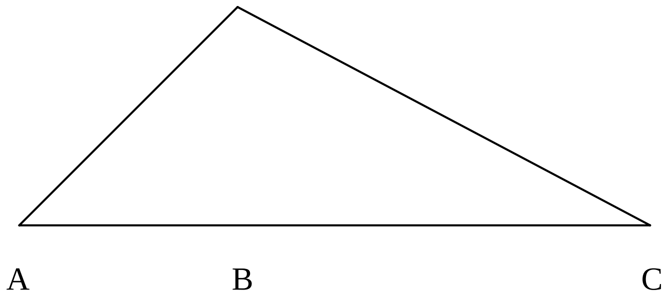
Question: The estimate of how much resource, such as supervisor time, is required by an activity is somewhat subjective. What is an easy way to test how sensitive project length is to these estimates?

Predicting Probability of Project Completion by Target

Simple critical path analysis assumes that we know activity times precisely. In reality, the time to complete a task is almost always different from our original estimate. Almost from the beginning of the PERT methodology, it was suggested that project planners provide three estimates for each task: a) an optimistic time, b) a most likely time, and c) a pessimistic time. Early PERT then had a very approximate procedure for estimating the distribution of the completion time of the entire project.

We use a more accurate approach. We somewhat arbitrarily assume that task durations have a triangular distribution with the smallest possible time being *A*, the mode being *B*, and the maximum possible time being *C*. An example triangular distribution is shown in Figure 7. When PERT was first introduced, a somewhat more obscure distribution, the Beta distribution, was used. Any convenient distribution could be used. The triangular is the simplest to explain in terms of most optimistic time, most likely time, and most pessimistic time.

Figure 7 A Triangular Distribution



Let us consider our original home building project. Below are optimistic, most likely, and pessimistic estimates for each task in the project.

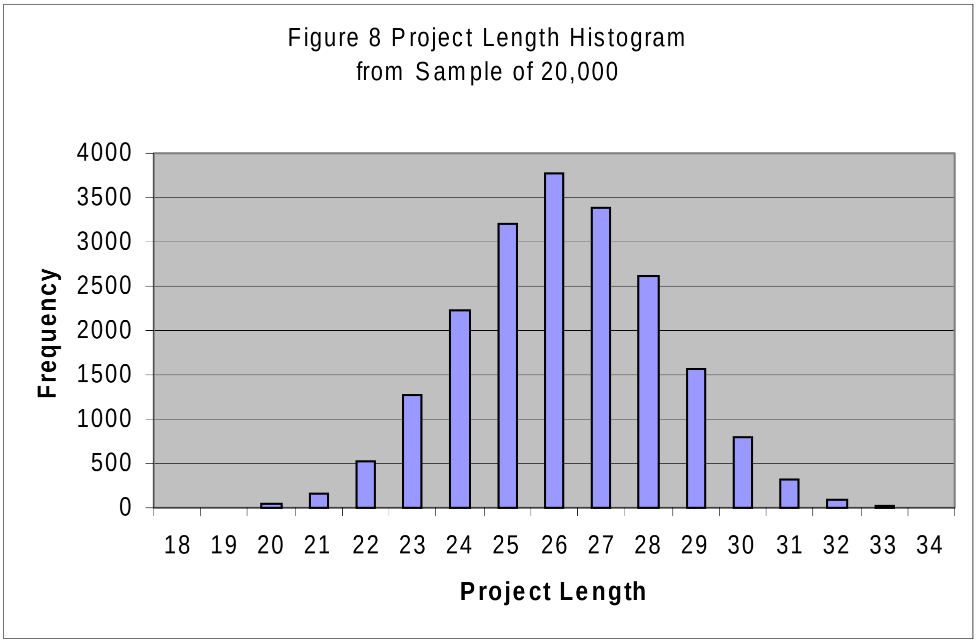
```
TASK,   A, B, C =
DIG     1  3  5
FOUND   2  4  6
JOISTS  2  4  6
POURB   1  3  5
WALLS   2  5  8
FLOOR   1  4  7
RAFTERS 1  3  5
ROUGH   3  6  9
ROOF    4  7 10
FINISH  2  4  6
SCAPE   3  9 15;
```

Notice for simplicity, we have assumed that the task time distributions are all symmetric, with mean equal to the single value used previously in the deterministic version of this project. We can get a distribution for the entire project time by repeatedly: a) choosing a random time for each task and b) computing the project length for the current choice of task times. This procedure is implemented using LINGO's *QRAND* function in the model pertrand.

The original, "one scenario" analysis concluded that the project would take 26 days. When task times are random, several questions come to mind. For example,

a) will the average or expected project time still be 26 days?
b) Given that the project length is random, what is the probability that in fact the project will in fact take 29 days or more?

Based on the same 20,000 simulations, the distribution of project length is shown in Figure 8. In answer to the questions, based on a sample of 20,000 projects, the average project length was 26.708 days. The probability that the project will complete in 29 days or more is 0.14.



Question:
Why might you expect the average project length to in fact be longer when individual tasks times are random, even though the individual tasks each have expected length equal to the original fixed time?