

## Chapter 2

# The PERT/CPM Technique

**Abstract** Completing a project on time and within budget is not an easy task. The project scheduling phase plays a central role in predicting both the time and cost aspects of a project. More precisely, it determines a timetable in order to be able to predict the expected time and cost of each individual activity.

In this chapter, the basic critical path calculations of a project schedule are highlighted and the fundamental concept of an activity network is presented. Throughout all chapters of Part I, it is assumed that a project is not subject to a limited amount of resources. The project is structured in a network to model the precedences between the various project activities. The basic concepts of project network analysis are outlined and the Program Evaluation and Review Technique (PERT) is discussed as an easy yet effective scheduling tool for projects with variability in the activity duration estimates.

### 2.1 Introduction

In this chapter, the basic concepts of the definition phase (Sect. 2.2) and the scheduling phase (Sect. 2.3) of the project life cycle are discussed. It is assumed that projects belong to the first quadrant of the project mapping matrix of Fig. 1.4 and hence are assumed to have no resource limits and a low level of uncertainty.

The chapter aims to give answers to fundamental questions, such as:

- What is the expected project finish date?
- How can precedence relations between activities be modeled in a network?
- What are the expected activity start and finish times?
- What is the effect of variability in activity time estimates on the project duration?

## 2.2 Project Definition Phase

In the definition phase of a project's life cycle, the organization defines the project objectives, the project specifications and requirements and the organization of the entire project. In doing so, the organization decides on how it is going to achieve all project objectives.

The Work Breakdown Structure (WBS) is a fundamental concept of the definition phase that, along with the Organizational Breakdown Structure (OBS), identifies the set of activities needed to achieve the project goal as well as the responsibilities of the project team for the various subparts of the project.

This information needs to be transformed into a network diagram that identifies a list of project activities and the technological links with the other activities. This project network is an easy and accessible tool for the critical path calculations to determine the earliest and latest activity start times of the scheduling phase.

### 2.2.1 WBS and OBS

The preparation of a Work Breakdown Structure (WBS) is an important step in managing and mastering the inherent complexity of the project. It involves the decomposition of major project deliverables into smaller, more manageable components until the deliverables are defined in sufficient detail to support development of project activities (PMBOK 2004). The WBS is a tool that defines the project and groups the project's discrete work elements to help organize and define the total work scope of the project. It provides the necessary framework for detailed cost estimation and control along with providing guidance for schedule development and control. Each descending level of the WBS represents an increased level of detailed definition of the project work.

The WBS is often displayed graphically as a hierarchical tree. It has multiple levels of detail, as displayed in Fig. 2.1.

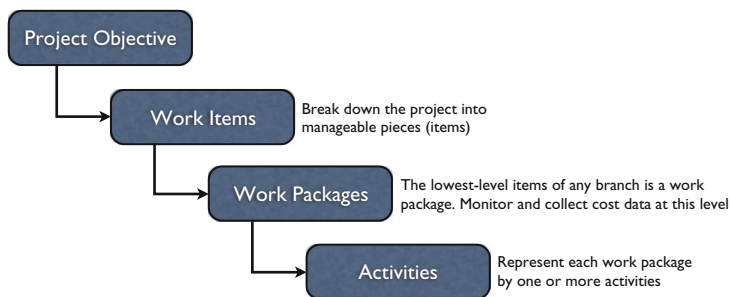


Fig. 2.1 Four levels of a Work Breakdown Structure

- Project objective: The project objective consists of a short description of the scope of the project. A careful scope definition is of crucial importance in project management.
- Work item: The project is broken down into manageable pieces (items) to be able to cope with the project complexity.
- Work package: The monitoring and collection of cost data often occurs at this level.
- Activity: The lowest level of the WBS, where the accuracy of cost, duration and resource estimates can be improved, and where the precedence relations can be incorporated.

The WBS is often used in conjunction with the Organizational Breakdown Structure (OBS). The OBS indicates the organizational relationships and is used as the framework for assigning work responsibilities. The WBS and the OBS are merged to create a Responsibility Assignment Matrix (RAM) for the project manager. The RAM displays the lower levels of both the WBS and the OBS and identifies specific responsibilities for specific project tasks. It is at this point that the project manager develops control accounts or work packages.

Figure 2.2 shows a graphical picture of a WBS/OBS conjunction and shows the RAM for a fictitious project. In the figure, the RAM uses the lowest level of the

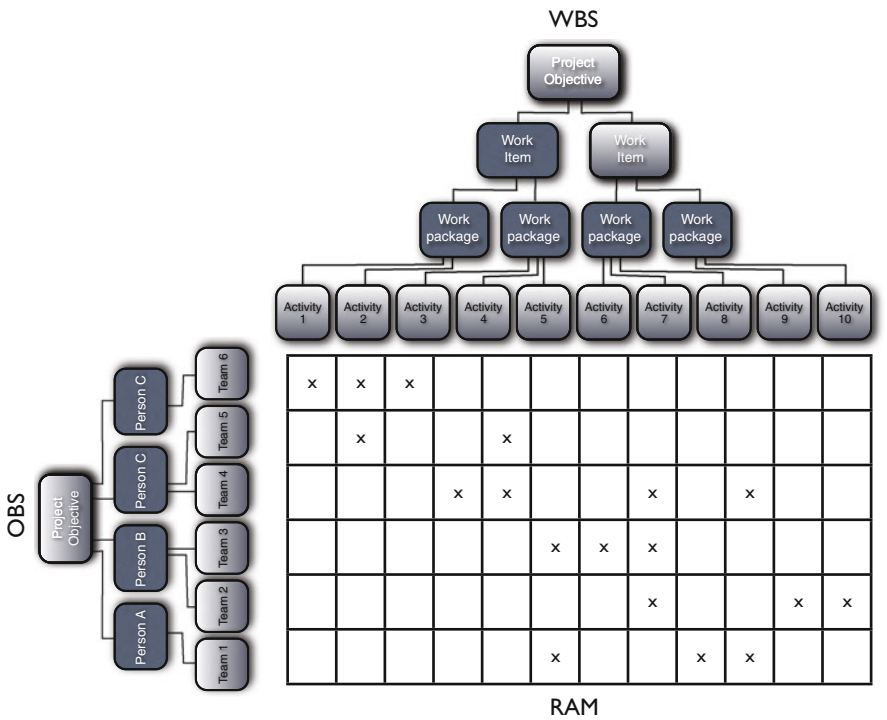


Fig. 2.2 A Responsibility Assignment Matrix (RAM)

WBS (activity level) and OBS and defines the specific person/department from the OBS assigned to be responsible for completing the activity from the WBS (indicated by an 'x'). Obviously, in practice, the responsibilities are often assigned to higher WBS levels (work package or work item level).

An illustration of a WBS is given in Chap. 4 of this book. The assignments and scheduling of resources from the OBS to the project activities is extensively discussed in the resource-constrained scheduling techniques of Chaps. 7 and 8.

### 2.2.2 Network Analysis

In order to construct a complete and detailed WBS, the work packages of a WBS need to be further subdivided into activities. In doing so, it might improve the level of detail and accuracy of cost, duration and resource estimates which serve as inputs for the construction of a project network and scheduling phase. Note that a clear distinction between the project definition phase and the project scheduling phase will be made throughout this chapter. The definition phase, which determines the list of activities, the precedence relations, possible resource requirements and the major milestones of the project, is different from the scheduling phase in the level of detail and the timing of project activities. Indeed, the scheduling phase aims at the determination of start and finish times of each activity of the project, and consequently, determines the milestones in detail. This can only be done after the construction of the network in the definition phase. Therefore, the activity description with the corresponding WBS-code and the estimates for its duration, cost and resource requirement are the main outputs of the definition phase and serve as inputs for the scheduling phase. In the latter phase, the earliest and latest possible start (and finish) time will be determined, given the technological precedence relations and limited resource constraints. The construction of a project schedule based on a project network with precedence relations is discussed in Sect. 2.3 of this chapter, while the introduction of resources in a project network is the topic of Part II of this book.

Many activities involve a logical sequence during execution. The links between the various activities to incorporate these logical sequences are called technological precedence relations. The annex *technological* is used to distinguish with the so-called *resource* relations, which will be introduced in Chaps. 7 and 8. Incorporating these technological links between any pair of activities is a first step in the construction of the project network. A network consists of nodes and arcs and incorporates all the activities and their technological precedence relations. A network can be seen as a graph  $G(N, A)$  where the set  $N$  is used to denote the set of nodes and  $A$  to denote the set of arcs. The network has a single start node and a single end node and is used as an input for the scheduling phase as discussed in Sect. 2.3. The set of activities of a project and their corresponding technological precedence relations can be displayed as a network using two formats: an activity-on-the-node (AoN)

and an activity-on-the-arc (AoA) representation. In the next subsections, these two formats are discussed in more detail.

### Activity-on-the-Arc (AoA)

In an AoA format, activities are displayed by means of arcs in the network. The nodes are events (or milestones) denoting the start and/or finish of a set of activities of the project. The technological link between activity  $i$  and activity  $j$  can be displayed as in Fig. 2.3. Since activities can be labeled with their corresponding start and end node event, it is said that activity (2,3) is a successor of activity (1,2) and activity (1,2) is a predecessor of activity (2,3).

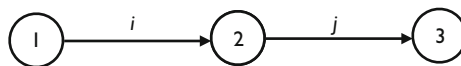
Moder et al. (1983) have suggested six rules to construct AoA networks, as follows:

1. Before any activity may begin, all activities preceding it must be completed.
2. Arrows imply logical precedence only. Neither the length nor its “compass” direction have any difference.
3. Event numbers must not be duplicated in a network.
4. Any two events may be directly connected by no more than one activity.
5. Networks may have only one initial event (with no predecessor) and only one terminal event (with no successor).
6. The introduction of dummy activities is often necessary to model all precedence relations.

Dummies are introduced for the unique identification of activities and/or for displaying certain precedence relations. These activities are represented by dashed arrows in the network and do not consume time nor resources.

Figure 2.4 displays an example project with a dummy arc to identify all activities in a unique way. The network contains two activities that can be performed in parallel (i.e. there is no technological precedence relation between the two activities). Rule 4 states that two events may not be connected by more than one activity to ensure the unique identification of each activity (both activities  $i$  and  $j$  can be labeled as activity (1,2)). Therefore, an extra dummy activity needs to be embedded in the project network, represented by the dashed arcs. In doing so, the network starts and ends with a single event node (rule 5) and each activity has been defined by a unique start/end event combination (rule 4).

Table 2.1 displays a list of project activities with each their immediate predecessors to illustrate the necessity of dummy arcs to incorporate all precedence relations. In Fig. 2.5, this activity information has been translated into an AoA network in



**Fig. 2.3** The AoA representation of the technological link between activities  $i$  and  $j$

two ways. Figure 2.5a has three dummies (D-E, G-J and I-J) while Fig. 2.5b only has one dummy activity (D-E). This single dummy activity is necessary to incorporate the precedence relation between activity 4 and its successor activity 8. However, the incorporation of dummy activities implies different possible alternative translations of the project data into a network and hence the project network is not unique.

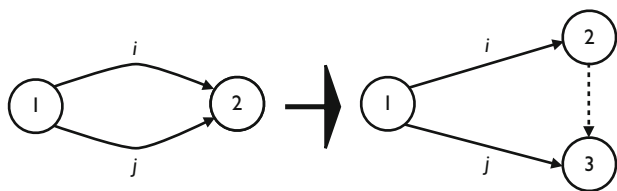


Fig. 2.4 Dummy arc for the unique identification of activities *i* and *j*

Table 2.1 List of activities with their immediate predecessors

Activity	Predecessors
1	—
2	1
3	1
4	2
5	2
6	3
7	4
8	4, 5
9	6
10	8
11	7, 9, 10

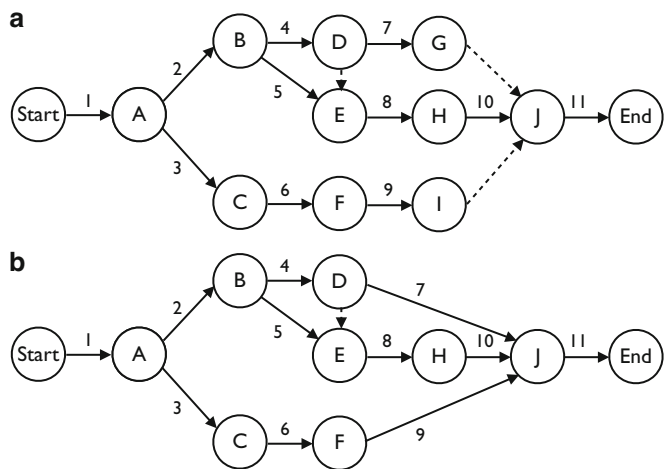


Fig. 2.5 Two AoA networks for the example project of Table 2.1

The introduction of dummy activities, which leads to different network representations of the same project, unnecessarily increases the project network complexity. Many researchers have focused on the development of (complex) algorithms to minimize the number of dummy activities in an AoA network. These algorithms are, due to their inherent complexity, outside the scope of this book. However, Wiest and Levy (1977) have presented some guidelines for reducing the number of dummy activities. Although these guidelines do not aim at minimizing the number of dummy activities in an AoA network, they can be very helpful in reducing superfluous dummy activities and hence, the project network complexity. The rules are as follows:

1. If a dummy node is the only activity emanating from its initial node, it can be removed.
2. If a dummy activity is the only activity going into its final node, it also can be removed.
3. If two (or more) activities have identical sets of predecessors (successors) then the two jobs should emanate from a single node connected to their predecessors (successors) by dummy activities.
4. Dummy jobs that show predecessor relations already implied by other activities may be removed as redundant.

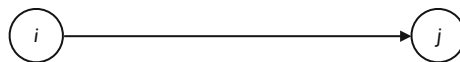
As an example, the first rule was used to reduce the number of dummy activities from 3 to 1 in Fig. 2.5.

### Activity-on-the-Node (AoN)

An AoN network displays the activities by nodes and precedence relations by arcs. Most commercial software tools rely on the activity-on-the-node format. The construction of an AoN network is very simple and is, in contrast to an AoA network, not subject to a set of rules. Dummy activities are not necessary, apart from a single initial start and a single end activity, which makes an AoN network always unique. The AoN representation of the technological link between activity  $i$  and activity  $j$  can be displayed as in Fig. 2.6.

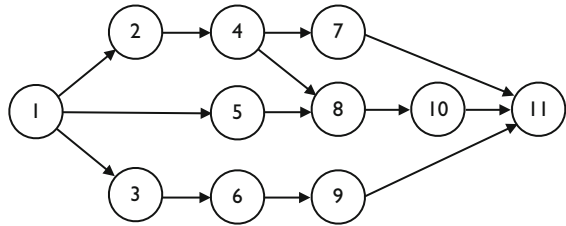
The AoN network for the project of Table 2.1 is given in Fig. 2.7. The three steps to follow in order to construct an AoN network are:

1. Draw a node for each network activity.
2. Draw an arc for each immediate precedence relation between two activities.
3. Possibly add a dummy start and dummy end node to force that the network begins with a single start activity and finishes with a single end activity.



**Fig. 2.6** The AoN representation of the technological link between activities  $i$  and  $j$

**Fig. 2.7** The AoN network for the example project of Table 2.1



Although both formats are only alternative ways to represent a project network, there might be reasons to have a preference towards one of the formats. Probably, many project managers will rely on the activity-on-the-node format, since this allows for the incorporation of generalized precedence relations (see Sect. 2.2.3), and is embedded in most resource allocation models of commercial software tools. However, some people might feel a preference for the activity-on-the-arc format, since the first scheduling principles (e.g. PERT (discussed at the end of this chapter) and CPM (Chap. 3)) have originally been developed for AoA networks. In the remainder of this book, we rely on the AoN format to represent project networks. More precisely, it is always assumed that a project is represented by an activity-on-the-node network where the set of nodes,  $N$ , represents activities and the set of arcs,  $A$ , represents the precedence constraints. The activities are numbered from 1 to  $n$  (i.e.  $|N| = n$ ), where node 1 and node  $n$  are used to denote the dummy start and dummy end activity, respectively. The dummy start activity is a predecessor for all activities in the network and is used to denote the start of the project. In a similar way, the dummy end activity  $n$  denotes the finish of the project and is a successor for all activities of the project. No further dummies are used in the AoN format.

### 2.2.3 Generalized Precedence Relations

In the previous subsections, technological precedence relations between project activities were implicitly assumed to be of the “Finish-Start” type. This section shows that these technological precedence relations can be extended to other types in three ways:

- Time-lag of precedence relations: zero or nonzero.
- Type of precedence relation: finish-start, finish-finish, start-start and start-finish.
- Time-lag requirement of a precedence relation: minimal or maximal.

In what follows, the three extensions are described in detail. Note that these precedence relations do not specify when activities have to start and end, but only describe possible relations between them. The former will be determined in the scheduling phase, while the latter is still subject to the definition phase. Indeed, while the definition phase determines *what* needs to be done in order to achieve the



project goals, the scheduling phase determines *when* all these necessary steps need to be performed.

### Time-Lag

A finish-start relation with a zero time-lag can be represented as follows:

$$FS_{ij}$$

*Activity j can only start after the finish of activity i*

A zero time-lag implies that the second activity *j* can start immediately after the finish of the first activity *i*, or later. It does not force the immediate start after the finish of the first activity, since the definition phase only describes the technological requirements and limitations and does not aim at the construction of a timetable.

A finish-start relation with a nonzero time-lag can be represented as follows:

$$FS_{ij} = n$$

*Activity j can only start n time periods after the finish of activity i*

### Type

The default precedence relation, finish-start, can be extended to other types of precedence relations, and can be used in combination with both zero and nonzero time-lags. The extensions are as follows:

$$SS_{ij} = n$$

*Activity j can only start n time periods after the start of activity i*

$$FF_{ij} = n$$

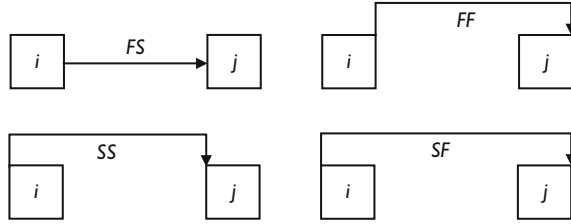
*Activity j can only finish n time periods after the finish of activity i*

$$SF_{ij} = n$$

*Activity j can only finish n time periods after the start of activity i*

Figure 2.8 graphically displays the four types of precedence relations between activities *i* and *j*.

**Fig. 2.8** Four types of precedence relations between activities  $i$  and  $j$



### Minimal/Maximal

In the previous sections, precedence relations were assumed to be minimal requirements between two activities. These technological requirements can be easily extended to maximal requirements.

A finish-start relation with a minimal time-lag of  $n$  can be represented as follows:

$$FS_{ij}^{\min} = n$$

*Activity  $j$  can only start  $n$  or more time periods after the finish of activity  $i$*

A finish-start relation with a maximal time-lag of  $n$  can be represented as follows:

$$FS_{ij}^{\max} = n$$

*Activity  $j$  can only start  $n$  or less time periods after the finish of activity  $i$*

Logically, the extension from a minimal to a maximal time-lag also holds for start-start, finish-finish and start-finish precedence relations. The precedence relations can, and often will be used in combination, as shown in an illustrative project network of Fig. 2.9. The numbers above each node denote the activity durations and the labels associated with the arcs refer to the generalized precedence relations.

Note that a maximal time-lag can be represented by a negative minimal time-lag in the opposite direction. Consequently, project networks with generalized precedence relations can be represented by cyclic networks. Figure 2.10 shows such a transformation from a  $FS_{ij}^{\max} = 3$  to a  $SF_{ij}^{\min} = -3$  relation. Activity  $j$  has to start maximum 3 time periods after the finish of activity  $i$ , which is exactly the same as specifying that activity  $i$  can only finish minimum  $-3$  time periods after the start of activity  $j$ .

The various time lags can be represented in a standardized form by transforming them to, for example, minimal start-start precedence relations  $l_{ij}$ , using the following transformation rules (Bartusch et al. 1988):

$$\begin{array}{ll}
 s_i + SS_{ij}^{\min} \leq s_j & \rightarrow s_i + l_{ij} \leq s_j \quad \text{with} \quad l_{ij} = SS_{ij}^{\min} \\
 s_i + SS_{ij}^{\max} \geq s_j & \rightarrow s_j + l_{ji} \leq s_i \quad \text{with} \quad l_{ji} = -SS_{ij}^{\max} \\
 s_i + SF_{ij}^{\min} \leq s_j + d_j & \rightarrow s_i + l_{ij} \leq s_j \quad \text{with} \quad l_{ij} = SF_{ij}^{\min} - d_j
 \end{array}$$

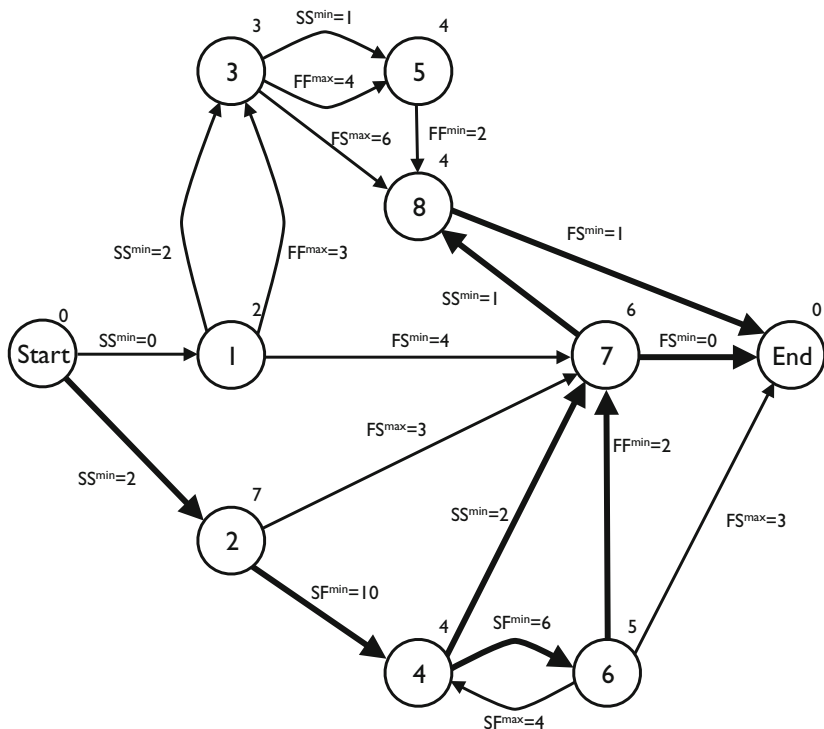


Fig. 2.9 An activity network with generalized precedence relations (Source: De Reyck 1998)

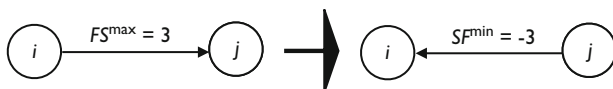


Fig. 2.10 The equivalence of minimal and maximal time lags

$$\begin{array}{ll}
 s_i + SF_{ij}^{\max} \geq s_j + d_j & \rightarrow s_j + l_{ji} \leq s_i \text{ with } l_{ji} = d_j - SF_{ij}^{\max} \\
 s_i + d_i + FS_{ij}^{\min} \leq s_j & \rightarrow s_i + l_{ij} \leq s_j \text{ with } l_{ij} = d_i + FS_{ij}^{\min} \\
 s_i + d_i + FS_{ij}^{\max} \geq s_j & \rightarrow s_j + l_{ji} \leq s_i \text{ with } l_{ji} = -d_i - FS_{ij}^{\max} \\
 s_i + d_i + FF_{ij}^{\min} \leq s_j + d_j & \rightarrow s_i + l_{ij} \leq s_j \text{ with } l_{ij} = d_i - d_j + FF_{ij}^{\min} \\
 s_i + d_i + FF_{ij}^{\max} \geq s_j + d_j & \rightarrow s_j + l_{ji} \leq s_i \text{ with } l_{ji} = d_j - d_i - FF_{ij}^{\max}
 \end{array}$$

with  $s_i$  the start time and  $d_i$  the estimated duration of activity  $i$ . If there is more than one time lag  $l_{ij}$  between two activities  $i$  and  $j$ , only the maximum time lag is retained. Figure 2.11 shows the project network of Fig. 2.9 after applying the transformation rules. The bold arcs in these two figures are used to display the so-called critical path, which will be explained in Sect. 2.3.2.

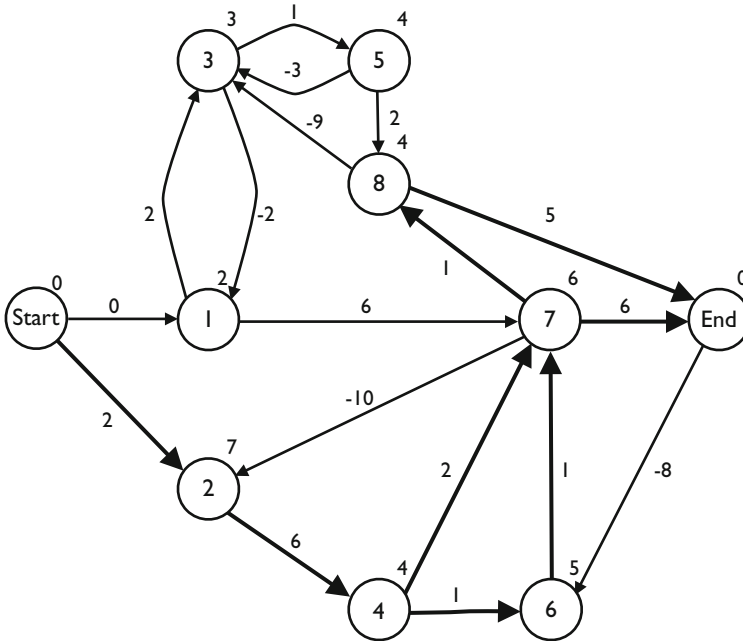


Fig. 2.11 The activity network of Fig. 2.9 with only minimal time lags

### 2.2.4 Other Constraint Types

The previous sections described the way how the technological relations between project activities can be incorporated in the project network. However, other project-specific requirements might show up in practice. A number of frequently occurring constraint types are:

- *Ready dates* imply earliest start or finish times on activities and hence force the activity to start or finish no earlier than the defined time instance. These constraints are known as ready start times (RST) or ready finish times (RFT).
- *Due dates* imply latest start or finish times on activities and force activities to start or finish no later than a predefined time instance. These constraints are referred to as due start times (DST) or due finish times (DFT).
- *Locked dates* imply a fixed time instance and force the activity to start or finish on a predefined time instance, known as locked start times (LST) or locked finish times (LFT).
- *And/Or* constraints allow activities to start when a predefined subset of predecessor activities has been finished.
- *Time-switch* constraints imply a project calendar to the activities to define shift workload patterns.

This nonexhaustive list can be easily extended, depending on the specific needs and requirements of the project. Many of these constraint types can be incorporated in an AoN project network by adding extra arcs in the network. Ready times, for example, can be incorporated in the network by adding an arc (dummy start,  $i$ ) of type  $SS_{1,i}^{\min} = r_i$  with  $r_i$  the ready time of activity  $i$ . In doing so, activity  $i$  can not start earlier than time instance  $r_i$ . A natural generalization of ordinary precedence constraints are so-called *and/or* precedence constraints. In the default *and* constraint, an activity must wait for all its predecessors while in an *or* constraint, an activity has to wait for at least one of its predecessors. A complete description of all possible constraint types is outside the scope of this chapter.

Time-switch constraints have been introduced as a logical extension to the traditional models in which it is assumed that an activity can start at any time after the finish of all its predecessors. To that purpose, two improvements over the traditional activity networks have been introduced by including two types of time constraints. *Time-window* constraints assume that an activity can only start within a specified time interval. *Time-schedule* constraints assume that an activity can only begin at one of an ordered schedule of beginning times. Moreover, these time constraints can be extended by treating time as a repeating cycle where each cycle consists of two categories: (1) some pairs of rest and work windows and (2) a leading number specifying the maximal number of times each pair should iterate. By incorporating these so-called time-switch constraints, activities are forced to start in a specific time interval and to be down in some specified rest interval. A typical example of a time-switch constraint is a regular working day: work intervals are time intervals between 9 and 12 a.m. and 1 and 5 p.m. while all the time outside these two intervals is denoted as rest intervals (Vanhoucke et al. 2002; Vanhoucke 2005).

A shift-pattern is very widely used by many companies and can be considered as a special type of time-switch constraints that force activities to start in a specific time period and which impose three different work/rest patterns:

- day-pattern: an activity can only be executed during day time, from Monday till Friday. This pattern may be imposed when many persons are involved in executing the activity.
- d&n-pattern: an activity can be executed during the day or night, from Monday till Friday. This pattern may be followed in situations where activities require only one person who has to control the execution of the activity once in a while.
- dnw-pattern: an activity can be in execution every day or night and also during the weekend. This may be the case for activities that do not require human intermission.

## 2.3 Project Scheduling Phase

The project network diagram and the activity time estimates made by the project manager during the definition phase will be used as inputs for the scheduling phase. The scheduling phase aims at the construction of a timetable to determine the

activity start and finish times and to determine a realistic total project duration within the limitations of the precedence relations and other constraint types. Although the minimization of the project lead time is often the most important objective during the scheduling phase, other scheduling objectives are often crucial from a practical point of view. In this chapter, only a time objective is taken into account. The extension to other scheduling objectives is the topic of Chaps. 7 and 8.

### ***2.3.1 Introduction to Scheduling***

Scheduling is an inexact process that tries to predict the future. More precisely, it aims at the construction of a timetable for the project where start and finish times are assigned to the individual project activities. Since activities are subject to several (precedence and resource-related) constraints, the construction of a schedule can be enormously complex. Indeed, project activities are precedence related and their execution may require the use of different types of resources (money, crew, equipment, ...). The scheduling objectives (often referred to as a measure of performance) may take many forms (minimizing project duration, minimizing project costs, maximizing project revenues, optimizing due date performance, ...).

The early endeavors of project management and scheduling date back to the development of the Gantt chart by Henry Gantt (1861–1919). This charting system for production scheduling formed the basis for two scheduling techniques, which were developed to assist in planning, managing and controlling complex organizations: the Critical path Method (CPM) and Program Evaluation and Review Technique (PERT). The Gantt chart is a horizontal-bar schedule showing activity start, duration, and completion.

The Critical Path Method was the discovery of M. R. Walker of E. I. Du Pont de Nemours and Co. and J. E. Kelly of Remington Rand, circa 1957. The first test was made in 1958, when CPM was applied to the construction of a new chemical plant. In March 1959, the method was applied to a maintenance shut-down at the Du Pont works in Louisville, Kentucky. The Program Evaluation and Review Technique was devised in 1958 for the POLARIS missile program by the program evaluation branch of the special projects office of the U.S. Navy. Due to the similarities of both techniques, they are often referred to as the PERT/CPM technique.

Thanks to the development of the personal computer, project scheduling algorithms started to shift to resource allocation models and the development of software with resource-constrained scheduling features (see Chaps. 7 and 8). From the 1990s on, numerous extensions of resource allocations and the development of tools (e.g. the CC/BM approach of Chap. 10) allowed the project manager to deal with both complexity and uncertainty at the same time (see the project mapping picture of Fig. 1.4).

In this chapter, the basic critical path calculations are discussed where it is assumed that projects are not subject to limited availability of resources and the scheduler follows a time-perspective scheduling objective. In Sect. 2.4, the Program

Evaluation and Review Technique is discussed, which extends the time objective of a schedule to probability calculations. In Part II of this book, the scheduling principles will be extended to projects with limited resource availabilities and scheduling objectives different from time minimization will be discussed.

2.3.2 Critical Path Calculations

Consider the data of Table 2.2 for a fictitious project with 12 nondummy activities (and a dummy start (1) and end (14) activity). The sets  $P_i$  and  $S_i$  are used to refer to the direct predecessors and successors of an activity  $i$ . Note that precedence relations will be of the  $FS_{ij}^{min} = 0$  type, unless indicated otherwise. All models and principles discussed throughout the chapters can be extended to generalized precedence relations between project activities, resulting in an increase in complexity but not in a fundamental difference in scheduling approach. Figure 2.12 displays the AoN network of Table 2.2, where the number above the node denotes the activity duration.

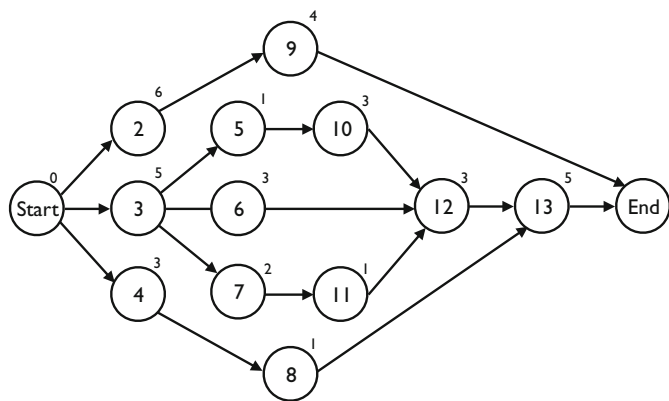
A path in a network can be defined as a series of connected activities from the start to the end of the project. All activities (and consequently, all paths) must be completed to finish the project. Table 2.3 enumerates all possible paths of the example network of Fig. 2.12, with their corresponding total duration.

The earliest possible completion time of the project is equal to the longest path in the network. This path, referred to as the *critical path*, determines the overall project duration. Care must be taken to keep these activities on schedule, since delays in any of these activities result in a violation of the entire project duration.

The clever reader immediately recognizes the basic principle underlying the *Theory Of Constraints* (TOC) introduced by Dr. Eliyahu M. Goldratt in his book

**Table 2.2** A fictitious project example with 12 nondummy activities

Activity	Predecessors	Duration (days)
1	—	0
2	1	6
3	1	5
4	1	3
5	3	1
6	3	3
7	3	2
8	4	1
9	2	4
10	5	3
11	7	1
12	6, 10, 11	3
13	8, 12	5
14	9, 13	0



**Fig. 2.12** The AoN example network of Table 2.2

**Table 2.3** Enumeration of all possible paths of the project of Table 2.2

Path	Duration
1 → 2 → 9 → 14	10
1 → 3 → 5 → 10 → 12 → 13 → 14	17
1 → 3 → 6 → 12 → 13 → 14	16
1 → 3 → 7 → 11 → 12 → 13 → 14	16
1 → 4 → 8 → 13 → 14	9

titled “The Goal”(Goldratt and Cox 1984), which is geared to help organizations continually achieve their goals. The main focus of this theory is to determine the most important constraint of a production system as the main driver of system performance and to give guidelines to protect this constraint in order to prevent loss of performance. Although the original book discussed the TOC in a production environment, it is a general management philosophy that can be applied to the basic project scheduling principles, amongst many others, discussed in this section. It basically consists of three steps, as follows:

1. What is your target/goal? The current scheduling objective is time.
2. What is the bottleneck constraint in your system? The critical path determines the target (time).
3. Protect the system constraint! In Chap. 5, a Schedule Risk Analysis is discussed as a tool to measure and understand the weakest parts (i.e. the constraints) of a project in order to protect them against unexpected events.

A detailed discussion of TOC is outside the scope of this book. In Chap. 10, this theory is used to add safety time in a resource-constrained project schedule as discussed in another book by Goldratt titled “Critical Chain” (Goldratt 1997).

It might be tempting to generate all possible paths of a project network in order to determine the longest path of a project. Unfortunately, the large amount of paths and consequently the required CPU-time to generate them render such a method



inapplicable for networks with a realistic size. Therefore, software tools rely on a three step procedure in order to detect the critical path of a network, as follows:

1. Calculate the earliest start schedule
2. Calculate the latest start schedule
3. Calculate the slack for each activity

### Earliest Start Schedule (ESS)

The earliest start  $es_i$  of each activity  $i$  can be calculated using forward calculations in the project network. The earliest start of an activity is equal to or larger than the earliest finish of all its predecessor activities. The earliest finish  $ef_i$  of an activity  $i$  is defined as its earliest start time increased with its duration estimate.

The earliest start times can be calculated using the following forward calculations, starting with the dummy start node 1:

$$es_1 = 0$$

$$es_j = \max(es_i + d_i | i \in P_j)$$

and the earliest finish times are given by:

$$ef_i = es_i + d_i$$

It is easy to verify that the earliest start times of the project activities of Table 2.2 are given by  $es_1 = 0, es_2 = 0, es_3 = 0, es_4 = 0, es_5 = 5, es_6 = 5, es_7 = 5, es_8 = 3, es_9 = 6, es_{10} = 6, es_{11} = 7, es_{12} = 9, es_{13} = 12$  and  $es_{14} = 17$ . The overall minimal project duration equals 17 time units.

### Latest Start Schedule (LSS)

The latest finish  $lf_i$  of each activity  $i$  can be calculated in an analogous way, using backward calculations, starting from the project deadline  $\delta_n$  at the dummy end node of the project. The latest finish of an activity is equal to or less than the latest start of all its successor activities. The latest start  $ls_i$  of an activity  $i$  is defined as its latest finish time decreased with its duration estimate.

The latest finish times can be calculated using the following backward calculations, starting with the dummy end node  $n$ :

$$lf_n = \delta_n$$

$$lf_i = \min(lf_j - d_j | j \in S_i)$$

and the latest start times are given by:

$$ls_i = lf_i - d_i$$

Given the project deadline of 17 time units, calculated as the earliest start of the end dummy activity in the previous step, the latest start times of each activity are given by  $ls_1 = 0$ ,  $ls_2 = 7$ ,  $ls_3 = 0$ ,  $ls_4 = 8$ ,  $ls_5 = 5$ ,  $ls_6 = 6$ ,  $ls_7 = 6$ ,  $ls_8 = 11$ ,  $ls_9 = 13$ ,  $ls_{10} = 6$ ,  $ls_{11} = 8$ ,  $ls_{12} = 9$ ,  $ls_{13} = 12$  and  $ls_{14} = 17$ .

### Activity Slack/Float

The amount of slack associated with each activity is used to denote the free time of each activity within the ESS and LSS. It denotes the amount of time each activity can be delayed without violating the entire project duration. The slack (or float) of activity  $i$  can be calculated as

$$ls_i - es_i = lf_i - ef_i$$

Activities with zero slack cannot be delayed without affecting the entire project duration and are called critical activities. Hence, the *critical path* consists of a path of critical activities and is given by activities 1, 3, 5, 10, 12, 13 and 14 in Table 2.4.

Activities that lie on the critical path cannot be delayed without delaying the entire project duration. Since time is an important objective in scheduling, the critical path is what the project manager has to focus on. It helps the manager to calculate the minimum length of time in which the project can be completed and which activities should be prioritized to complete the project within its deadline. In order to finish a project on time, the critical path calculations help the project manager to focus on the essential activities to which attention and resources should be devoted. It gives an effective basis for the scheduling and monitoring of progress.

**Table 2.4** The slack of the activities of the example project of Fig. 2.12

Activity	Slack
1	0
2	7
3	0
4	8
5	0
6	1
7	1
8	8
9	7
10	0
11	1
12	0
13	0
14	0

Gantt Charts

The Gantt chart is named after its originator Henry Gantt and displays a timetable for each activity of the project. Each activity is shown as a block or bar and drawn to scale in time. The timescale is usually drawn horizontally while the different activities are displayed on the vertical axis. This chart is used for scheduling and is often used in conjunction with the project network to show the technological dependencies between activities. Figure 2.13 shows an ESS Gantt chart of Table 2.2 along with the activity slack. The bars represent the earliest start and finish times of each activity of the example project. The gray lines following the activity bars represent the activity slack, and hence, shifting activities towards the end of these gray bars results in the corresponding LSS. Activities without slack (i.e. activities 1, 3, 5, 10, 12, 13 and 14) belong to the critical path and need the attention of the project manager.

Note that the introduction of generalized precedence relations (see previous chapter) might increase the complexity to find the critical path, but involves no fundamental difference in scheduling. However, some anomalies can occur when introducing these generalized precedence relations. Assume that, for illustrative purposes, the precedence relations between activity 5 and 10 and activity 10 and 12 change to a finish-finish and start-start relation, respectively, both with a minimal time-lag of 3 time units, i.e.  $FF_{5,10}^{\min} = 3$  and  $SS_{10,12}^{\min} = 3$ . It is easy to verify that the critical path remains unchanged and still determines the entire project duration of 17 time units. However, the introduction of generalized precedence relations leads to counterintuitive results and sheds a new light on the philosophy of the critical path, activity crashing (see Chap. 3) and/or the effects of delays in critical activities. Indeed, decreasing the duration of the critical activity 10 from 3 time units to 1 time unit results in an increase of the project duration with 2 time units, as shown in Fig. 2.14 (activity 10 starts 2 time units later than in the schedule of Fig. 2.13). Hence, in the presence of generalized precedence relations, the general

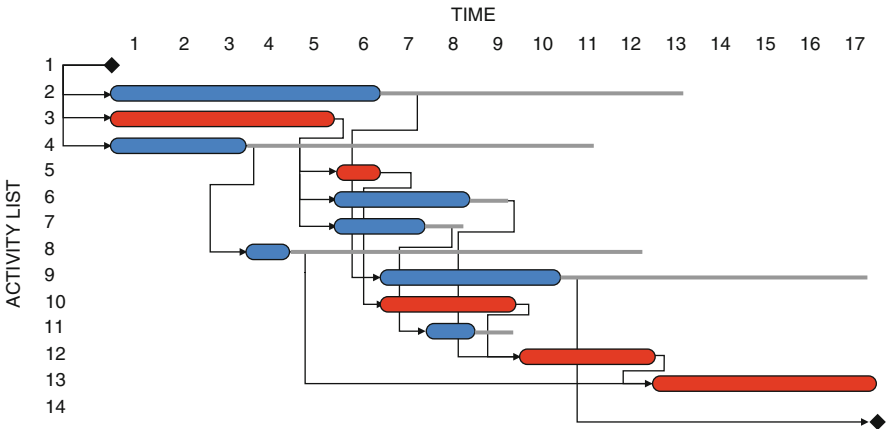
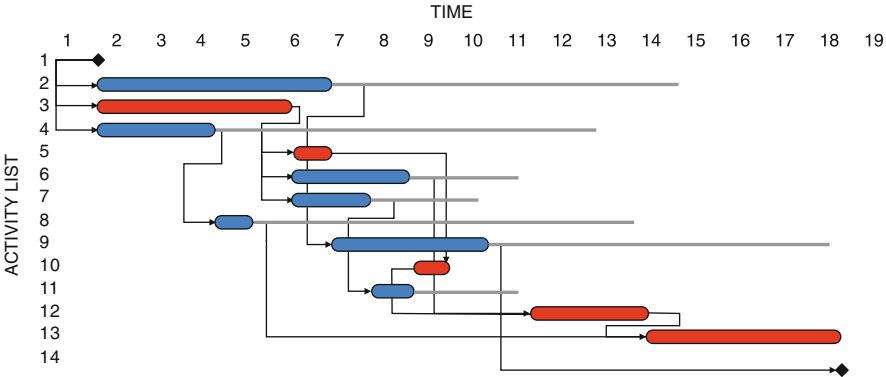


Fig. 2.13 The ESS Gantt chart of the example project and the activity slack



**Fig. 2.14** The modified ESS Gantt chart of the example project and the activity slack

rule “increasing the duration of a critical activity (or delaying this activity) results in an increase of the project duration” is no longer applicable. Instead, decreasing the duration of a critical activity sometimes results in an increase of the project duration!

**Resource Charts**

In the previous sections, it is assumed that project activities do not require resources during their execution (or alternatively, the assumption is that the resources are unlimited in availability). In practice, activities in progress need resources that are limited in availability. These resources have been classified in two basic categories, renewable and nonrenewable resources, and will be discussed in further chapters.

In Part II of this book, resource charts will show that the presence of resources under a limited availability will lead to changes in the project schedule. The resource charts, which extend the Gantt charts to the presence of renewable resources, will modify the activity start times and will extend the critical path concept into a so-called “critical chain” approach.

**2.4 Program Evaluation and Review Technique (PERT)**

The previous sections described the critical path calculations that form the basis of both the PERT and CPM technique. Due to the strong similarities between the two scheduling techniques, it is often referred to as the PERT/CPM technique. However, both techniques have their own characteristics and need further explanation. In the following subsections, the PERT technique is described, which aims at the construction of a precedence feasible schedule in the absence of resources. The

details of the critical path method (CPM) are reserved for Chap. 3, where the construction of a precedence feasible schedule with nonrenewable resources is discussed.

### 2.4.1 Three Activity Duration Estimates

In the previous section, it was assumed that the activity duration estimates, and the derived values for the earliest start, latest start, earliest finish and latest finish were all deterministic. In reality, this is seldom true and durations are often not known in advance. PERT has extended this deterministic approach in the face of uncertainty about activity times, and employs a special formula for estimating activity durations. The approach of PERT assumes that the activity duration estimates are done by someone who is familiar with the activity, and has enough insight in the characteristics of the activity. Hence, the technique requires three duration estimates for each individual activity, as follows:

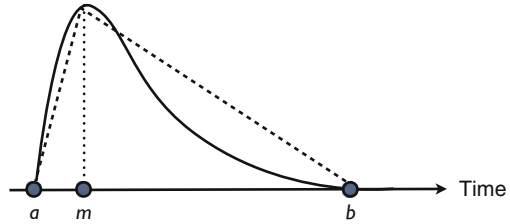
- Optimistic time estimate: This is the shortest possible time in which the activity can be completed, and assumes that everything has to go perfect.
- Realistic time estimate: This is the most likely time in which the activity can be completed under normal circumstances.
- Pessimistic time estimate: This is the longest possible time the activity might need, and assumes a worst-case scenario.

Table 2.5 displays the three time estimates for the activities of the example project of Fig. 2.12. The optimistic time estimate is denoted by  $a$ , the pessimistic time estimate is denoted by  $b$  and  $m$  is used to refer to the realistic time estimate.

**Table 2.5** Three time estimates for the activities of the project of Fig. 2.12

Activity	Optimistic $a$	Realistic $m$	Pessimistic $b$	$t$	$\sigma$
1	0	0	0	0	0
2	3	5	13	6	1.67
3	2	3.5	14	5	2
4	1	3	5	3	0.67
5	1	1	1	1	0
6	1	2	9	3	1.33
7	1	2	3	2	0.33
8	1	1	1	1	0
9	1	4	7	4	1
10	1	3	5	3	0.67
11	0.5	1	1.5	1	0.17
12	1	2.5	7	3	1
13	1	4	13	5	2
14	0	0	0	0	0

**Fig. 2.15** A beta distribution used to express activity duration variability in PERT



PERT assumes that each activity duration is a random variable between two extreme values (i.e.  $a$  and  $b$ ) and follows a beta probability distribution. A typical beta distribution function and its triangular approximation is shown in Fig. 2.15. Note that the difference between  $m$  and  $b$  is often, but not necessarily larger than the difference between  $a$  and  $m$  to express positive skewness.

The expected time  $t$  of a beta distribution can be approximated by the weighted average that sums to one, as follows:

$$t = \frac{a + 4m + b}{6} \quad (2.1)$$

The standard deviation of an activity duration, which serves as a measure for risk, can be calculated using the philosophy of a three standard deviations interval, as follows:

$$\sigma = \frac{b - a}{6} \quad (2.2)$$

This calculation is indeed inspired by a so-called three sigma interval for the normal distribution. A three-sigma interval for the normal distribution is the interval between the average minus three times the standard deviation and the average plus three times the standard deviation, as exactly 99.73% of the observations lie in that interval. By calculating the standard deviation based on a similar interval as given in Eq. 2.2, PERT assumes that almost none of the observations (i.e. real activity durations) will lie outside the  $[a, b]$  range.

### 2.4.2 Probability of Project Completion

The assumption that activity durations are random variables implies that the entire project duration is also a random variable. Hence, the entire project duration differs from the deterministic or expected project duration, due to the variability in the activity times as measured by its standard deviation. The PERT analysis allows to provide the following information:

- The expected entire project duration and the critical path.
- The probability to complete the project within a specified deadline.
- The deadline before which the project can be completed with a certain probability.

The PERT analysis calculates the expected critical path based on the expected duration of each activity. In the example project, the expected critical path  $E(T)$  is equal to 17 time units and consists of the activities 1, 3, 5, 10, 12, 13 and 14. Since each activity is assumed to be a random variable following a beta probability distribution, the total expected duration  $E(T)$  is also a random variable with a known distribution. This known distribution can be derived using the well-known *central limit theorem*.

The central limit theorem states that, given a distribution with an average  $E(T)$  and variance  $Var(T)$ , the sampling distribution of the mean approaches a normal distribution with an average  $E(T)$  and a variance  $Var(T)/n$  as  $n$ , the sample size, increases.

In the example project, the sample consists of the expected critical activities 1, 3, 5, 10, 12, 13 and 14 with each an average duration calculated earlier. Although the CLT as described above is formulated for the sampling distribution of the *mean*, the project completion time is simply the sum of the expected activity times for the critical path activities and hence, a total average duration of 17 time units can be calculated. Similarly, a total variance<sup>1</sup> that can be calculated as  $Var(T) = 2^2 + 0^2 + 0.67^2 + 1^2 + 2^2 = 9.44$  and, consequently, the standard deviation equals  $\sqrt{9.44} = 3.07$ .

Consequently, the example project follows a normal distribution with an average total duration of 17 time units and a standard deviation of 3.07, i.e.  $N(17; 3.07)$ . Using normal tables, or the well-known “normdist” or “norminv” functions in Microsoft Excel, it is easy to verify the calculations below.

**Probabilities:** The probability that the example project has a total duration less than or equal to 20 time units equals

$$\begin{aligned} P(T \leq 20) &= P\left(\frac{T - 17}{3.07} \leq \frac{20 - 17}{3.07}\right) \\ &= P(z \leq 0.976) \\ &= 83.55\% \end{aligned}$$

with  $z$  the symbol for the standardized value of the normal distribution.

**Percentiles:** The project duration  $T$  with a risk of exceeding of 10% is equal to the 90th percentile of the  $N(17; 3.07)$  normal distribution and can be calculated as follows:

$$\begin{aligned} 90\% \text{ percentile} \rightarrow z &= 1.28 = \frac{T - 17}{3.07} \\ \rightarrow T &= 20.9 \text{ time units (e.g. weeks)} \end{aligned}$$

---

<sup>1</sup>Only activities on the expected critical path are taken into account.

**Confidence Intervals:** The project will have a total duration between approximately 10.8 and 23.1 time units with a probability of 95%, which can be calculated as a  $2\sigma$  interval for the normal distribution. A more detailed statistical explanation can be found in any statistics handbook and is outside the scope of this book.

### 2.4.3 *Beyond PERT*

Despite the relevance of the PERT planning concept, the technique has often been criticized in literature. The PERT analysis implicitly assumes that all activities that are not on the critical path may be ignored by setting the activity durations to their average values. In realistic settings, projects have multiple critical paths instead of a single unique critical path. Moreover, in the stochastic setting, every noncritical path has the potential to become critical and hence the critical path would be the maximum of a set of possible critical paths. It is also assumed that the activity durations are independent random variables while in reality they can be dependent. These strict assumptions might lead to inaccuracies and has been the subject of a lot of research. In Chap. 5, the PERT technique is extended to Monte-Carlo simulation analyses, which allows to analyze the distribution of the critical path without the restricted PERT assumptions. For an overview of the pitfalls of making traditional PERT assumptions, the reader is referred to Elmaghraby (1977).

## 2.5 Conclusion

This chapter outlined the basic concepts of the definition and scheduling phases of a project's life cycle. The definition phase can be considered as the “what” phase since its main target is to determine what needs to be done in order to reach the project goal. It mainly consists of determining the main set of activities and precedence relations (modeled in a project network) and the responsibilities of the various project subparts. The scheduling phase, the “when” phase, constructs a timetable for the various project activities within a certain scheduling objective (which is assumed to be a time minimization objective in the current chapter), leading to a deterministic forecast of the earliest and latest activity start times and their slack within the minimal project duration.

In an attempt to add stochastic elements to the deterministic project schedule, the basic concepts of the PERT technique and an illustrative example have been discussed in the current chapter as a valuable tool to deal with a (low) degree of uncertainty in activity estimates. Although the basic concepts of the critical path scheduling approach have been outlined in this chapter, an overview of the main characteristics of the Critical Path Method (CPM), such as the activity time/cost trade-off function and the activity crashing possibility, is the subject of Chap. 3.



It should be noted that the usefulness of these scheduling methods should be put into the right perspective. Despite the relevance of introducing activity estimate variability in the project schedule under strict assumptions, the PERT principles have their inaccuracies and potential pitfalls. Consequently, the techniques discussed in this chapter are classified in the first quadrant (low uncertainty/low complexity) of the project mapping picture of Fig. 1.4. A project scheduling setting with higher degree of uncertainty is the topic of Chap.5. A higher degree of complexity in project scheduling is mainly due to the introduction of a limited resource availability, which will be discussed in Part II of this book.

<http://www.springer.com/978-3-642-40437-5>

Project Management with Dynamic Scheduling  
Baseline Scheduling, Risk Analysis and Project Control

Vanhoucke, M.

2013, XVIII, 318 p. 123 illus., Hardcover

ISBN: 978-3-642-40437-5