Software Project Management

Durga Prasad Mohapatra
Professor
CSE Deptt.
NIT Rourkela

Project Scheduling cont…

Formulating a Network Model

The first stage in creating a network model is to represent the activities and their interrelationships as a graph". In activity-on-node we do this by representing activities as nodes (boxes) in the graph – the lines between nodes represent dependencies.

Constructing precedence networks

Before we look at how networks are used, it is worth spending a few moments considering some rules for their construction.

- A project network should have only one start node
- A project network should have only one end node
- A node has duration
- Links normally have no duration
- Precedents are the immediate preceding activities
- Time moves from left to right
- A network may not contain loops
- A network should not contain dangles

A project network should have only one start node

- Although it is logically possible to draw a network with more than one starting node, it is undesirable to do so as it is a potential source of confusion.
- In such cases (for example, where more than one activity can start immediately the project starts) it is normal to invent a 'start' activity which has zero duration but may have an actual start date.

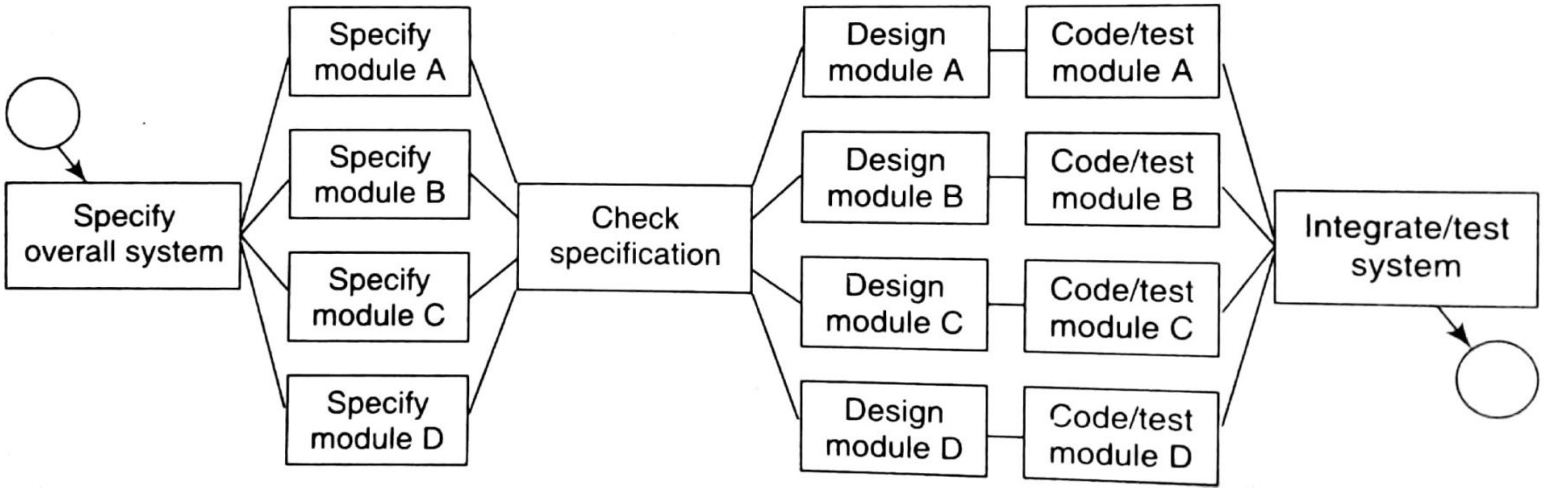
A project network should have only one end node

- The end node designates the completion of the project and a project may finish only once! Although it is possible to draw a network with more than one end node, it will almost certainly lead to confusion if this is done.
- Where the completion of a project depends upon more than one final activity it is normal to invent a 'finish' activity.

A node has duration

- A node represents an activity and, in general, activities take time to execute. Notice, however, that the network in the figure does not contain any reference to durations.
- This network drawing merely represents the logic of the project – the rules governing the order in which activities are to be carried out.

The IOE annual maintenance contracts project activity network fragment with a checkpoint activity added



Links normally have no duration

- Links represent the relationships between activities. In the figure installation cannot start until program testing is complete. Program testing cannot start until both coding and data take-on have been completed.

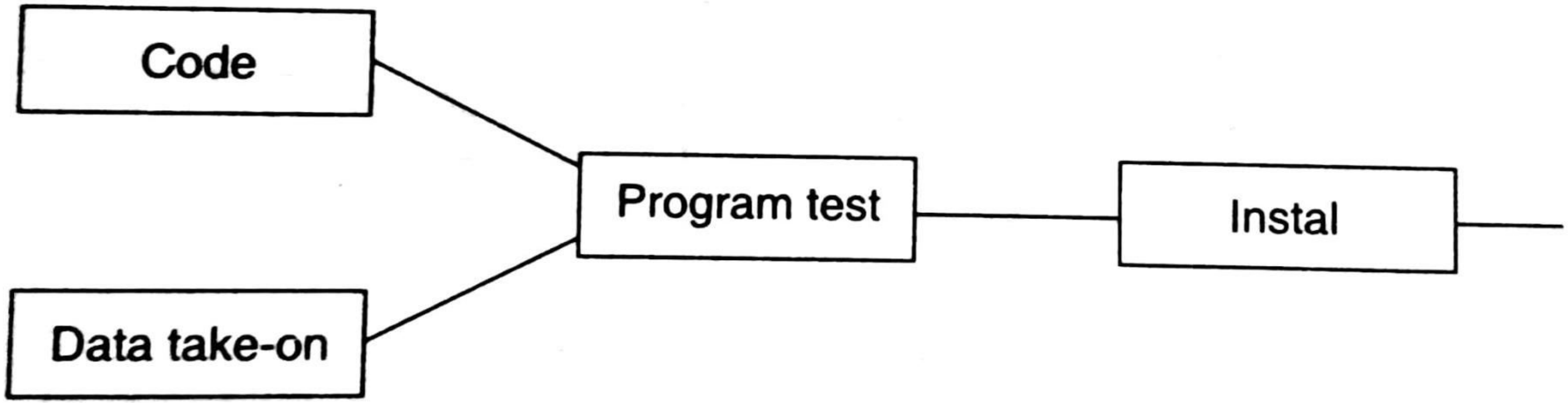


Figure: fragment of a precedence network

Precedents are the immediate preceding activities

- In the next figure, the activity Program test' cannot start until both ‘Code' and 'Data take-on' have been completed and activity 'Instal' cannot start until ‘Program test’ has finished.
- 'Code' and 'Data take-on‘ can therefore be said to be precedents of Program test', and 'Program test' is a precedent of 'Instal”.
- Note that we do not speak of 'Code' and 'Data take-on' as precedents of 'Instal' – that relationship is implicit in the previous statement.

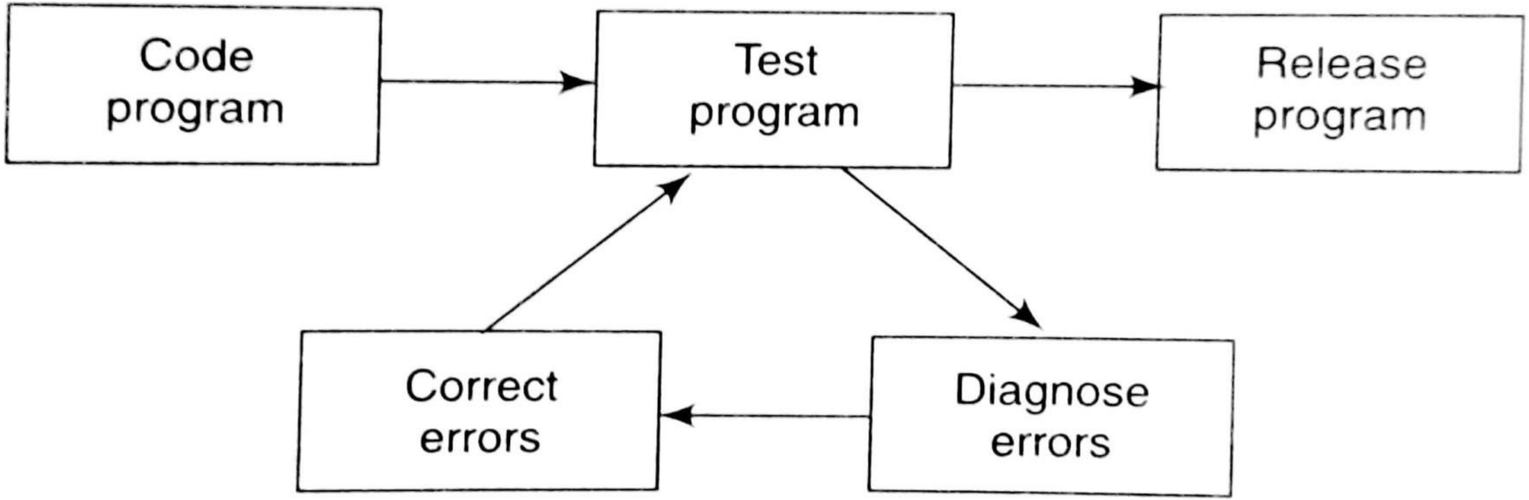
Time moves from left to right

- If at all possible, networks are drawn so that time moves from left to right.
- It is rare that this convention needs to be flouted but some people add arrows to the lines to give a stronger visual indication of the time flow of the project.

A network may not contain loops

- The next figure demonstrates a loop in a network. A loop is an error in that represents a situation that cannot occur in practice.
- While loops, in the sense of iteration, may occur in practice, they cannot be directly represented in a project network.
- Note that the logic of the figure suggests that program testing cannot start until the errors have been corrected.

A loop represents an impossible sequence



A network may not contain loops cont…

- If we know the number of times we expect to repeat a set of activities, a test-diagnose--correct sequence, for example, then we can draw that set of activities as a straight sequence, repeating it the appropriate number of times.
- If we do not know how many times a sequence is going to be repeated then we cannot calculate the duration of the project unless we adopt an alternative strategy such as redefining the complete sequence as a single activity and estimating how long it will take to complete it.
- Although it is easy to see the loop in this simple network fragment, very large networks can easily contain complex loops which are difficult to spot when they are initially constructed. Fortunately, all network planning applications will detect loops and generate error messages when they are found.

A network should not contain dangles

- A dangling activity such as 'Write user manual in the figure should not exist as it is likely to lead to errors in subsequent analysis.
- Indeed, in many cases dangling activities indicate errors in logic when activities are added as an afterthought.
- If, in the figure, we mean to indicate that the project is complete once the software has been installed and the user manual written then we should redraw the network with a final completion activity - which, at least in this case, is probably a more accurate representation of what should happen.
- The redrawn network is shown in the next figure.

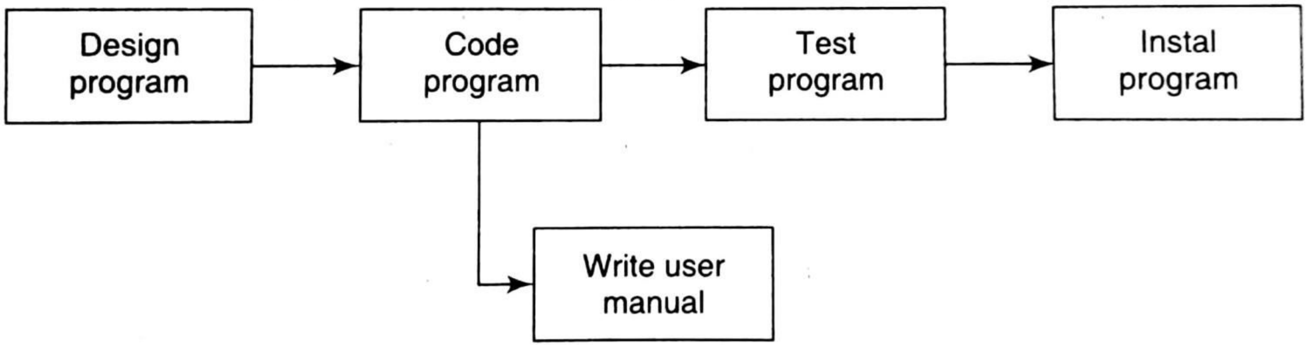


Figure: a dangle

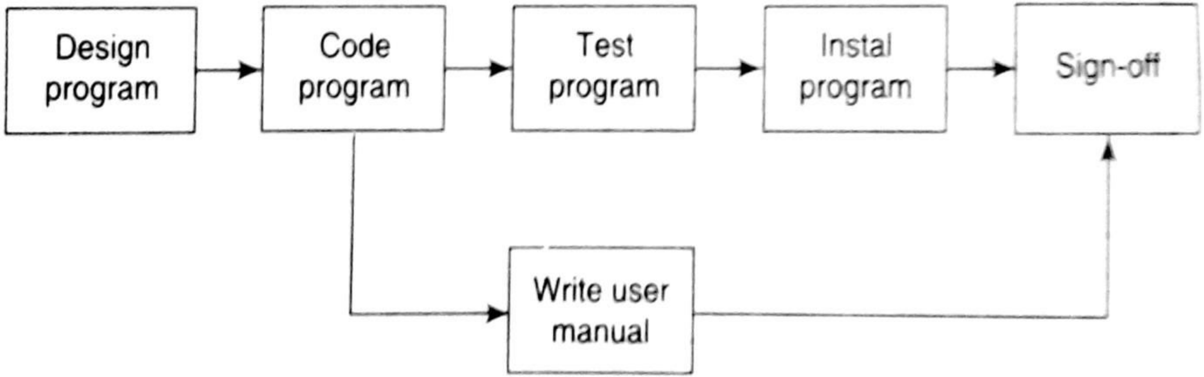


Figure: resolving the dangle

Representing lagged activities

- We might come across situations where we wish to undertake two activities in parallel so long as there is a lag between the two. We might wish to document amendments to a program as it is being tested - particularly if evaluating a prototype. In such a case we could designate an activity test and document amendments'. This would, however, make it impossible to show that amendment recording could start, say, one day after testing had begun and finish a little after the completion of testing.
- Where activities can occur in parallel with a time lag between them, we represent the lag with a duration on the linking arrow as shown in the next figure. This indicates that documenting amendments can start one day after the start of prototype testing and will be completed two days after prototype testing is completed.

Representing lagged activities

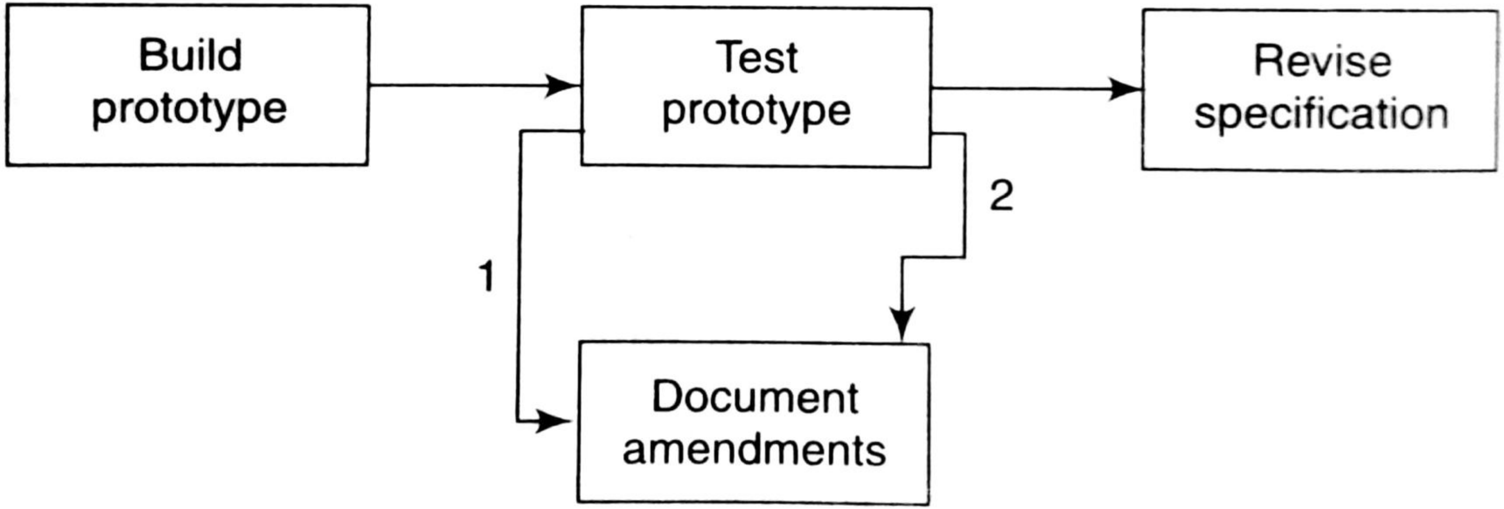


Figure: Activity Network indicating lags

Hammock activities

- Hammock activities are activities which, in themselves, have zero duration but are assumed to start at the same time as the first 'hammocked' activity and to end at the same time as the last one.
- They are normally used for representing overhead costs or other resources that will be incurred or used at a constant rate over the duration of a set of activities.

Another way of labeling activities

<table>
<tr><td>Earliest start</td><td>Duration</td><td>Earliest finish</td></tr>
<tr><td colspan="3">Activity label, activity description</td></tr>
<tr><td>Latest start</td><td>Float</td><td>Latest finish</td></tr>
</table>

Example

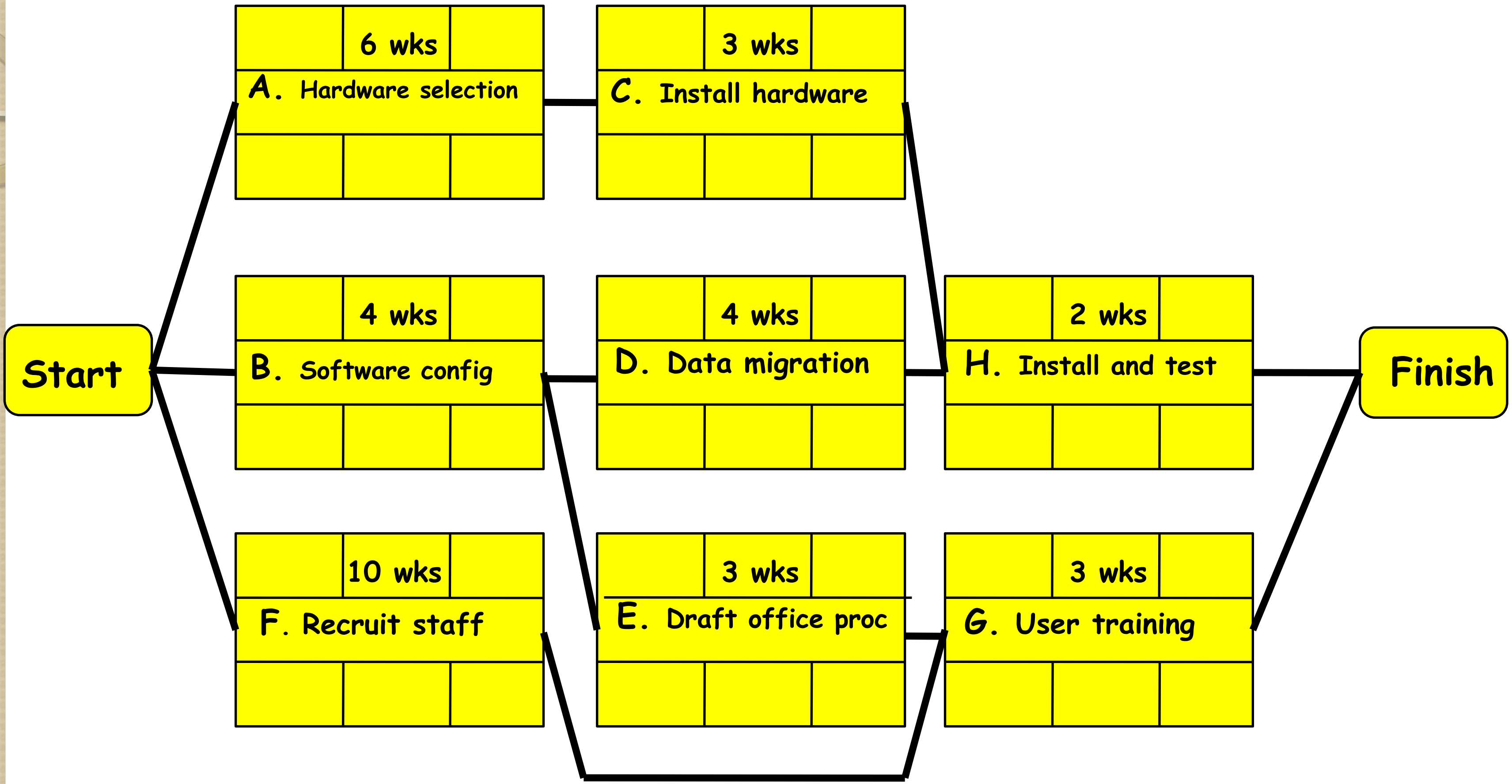
Find the critical activities and the critical path for the following example.

Activity Label	Activity Name	Duration (Weeks)	Precedence
A	Hardware Selection	6	---
B	System configuration	4	---
C	Install hardware	3	A
D	Data migration	4	B
E	Draft office procedures	3	B
F	Recruit staff	10	---
G	User training	3	E,F
H	Install and test	2	C,D

Steps

- First Draw Task Network
- Compute ES, EF, LS and LF
- Compute the slack time (float time) for each activity
- Identify the critical activities, I,e, activities for which slack time is zero.
- Identify the critical path, i.e. the which contains only the critical activities.
- There may be more than one critical path.

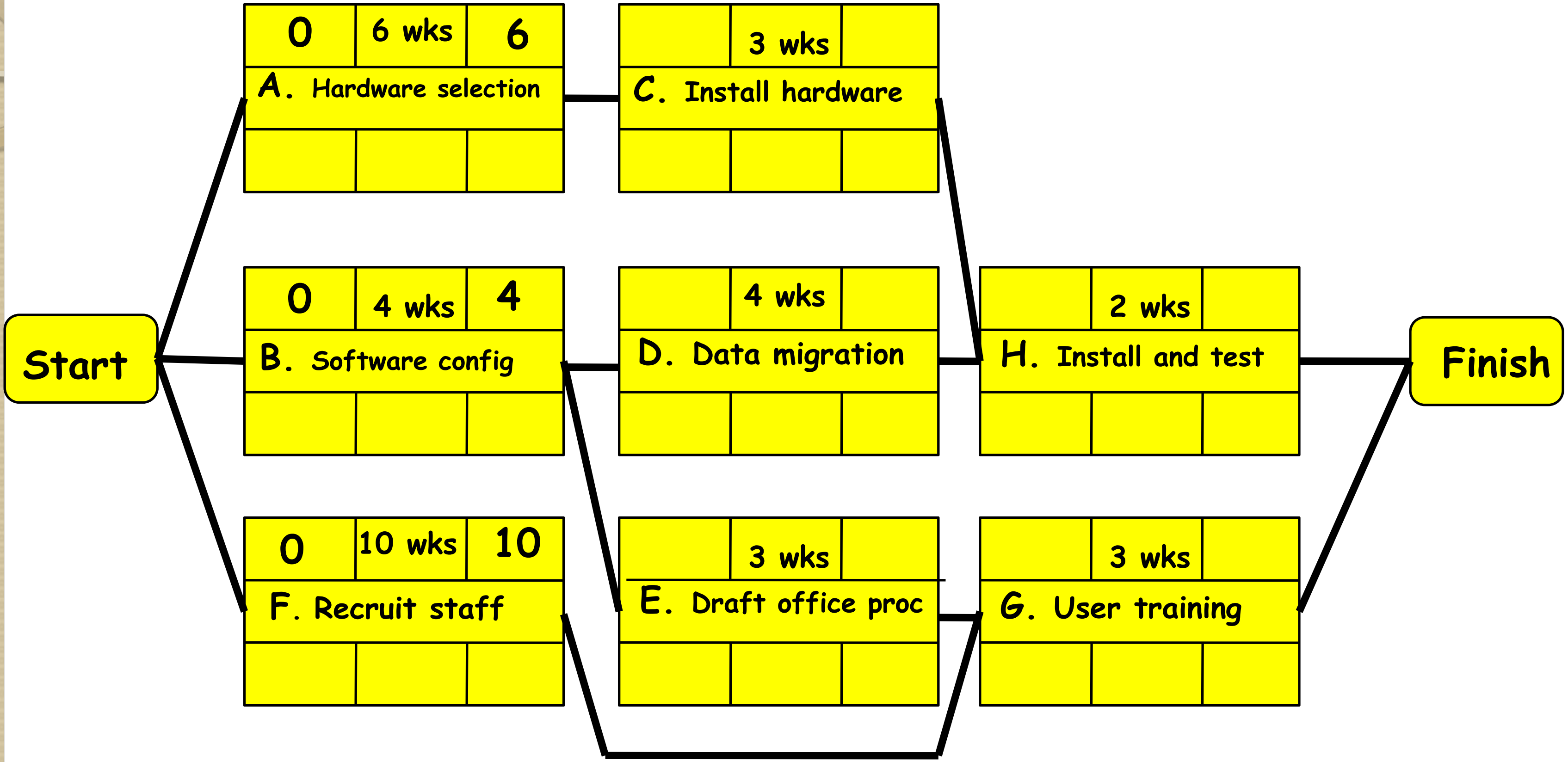
Activity Network



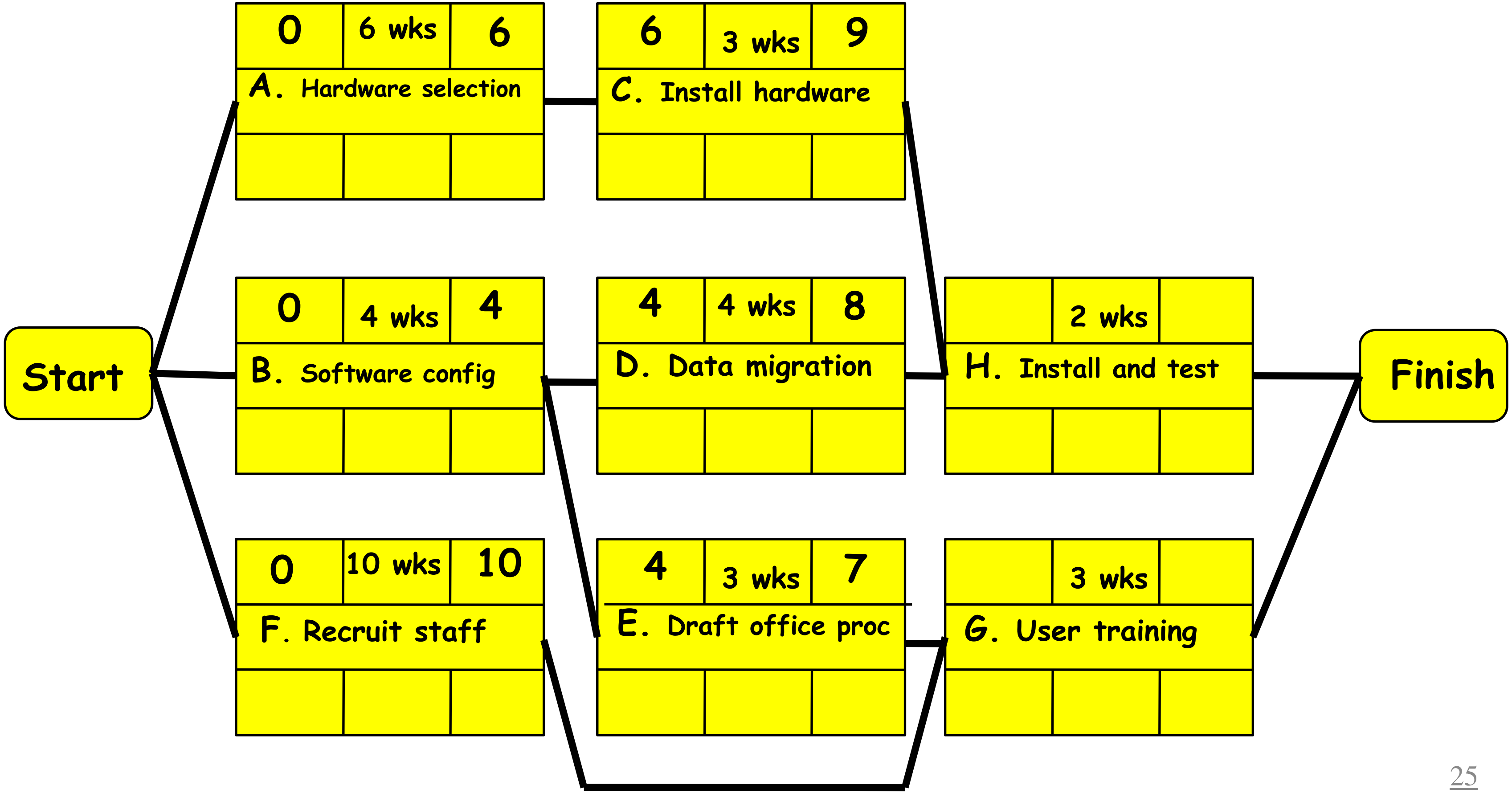
Forward Pass

- The forward pass is carried out to calculate the earliest dates on which each activity may be started and completed.
- Rule:
  - The earliest start date for an activity is the earliest finish date for the preceding activity.
  - Where there is more than one immediately preceding activity we take the latest of the earliest finish dates for those activities.

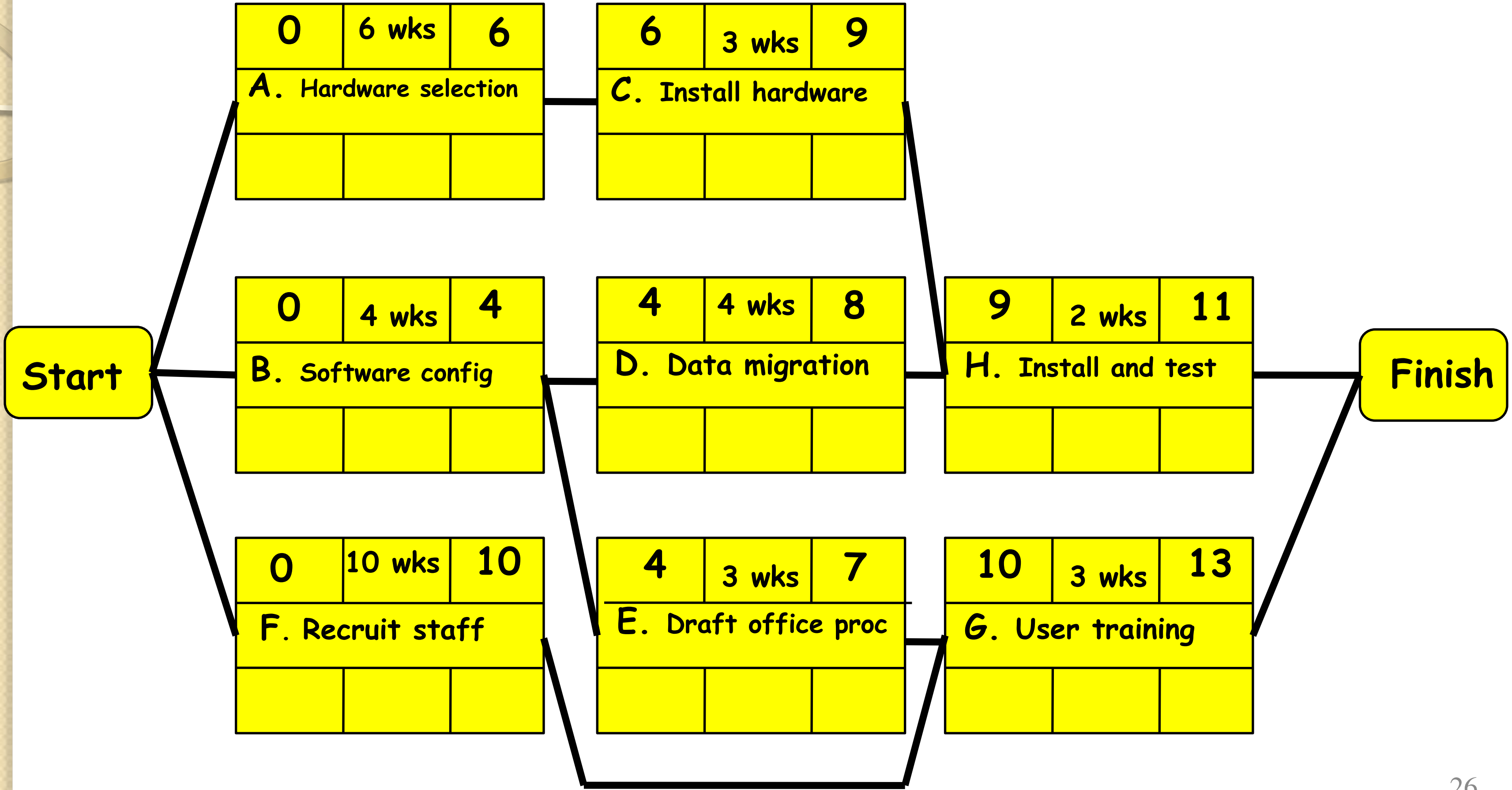
After Forward Pass Step 1



After Forward Pass Step 2



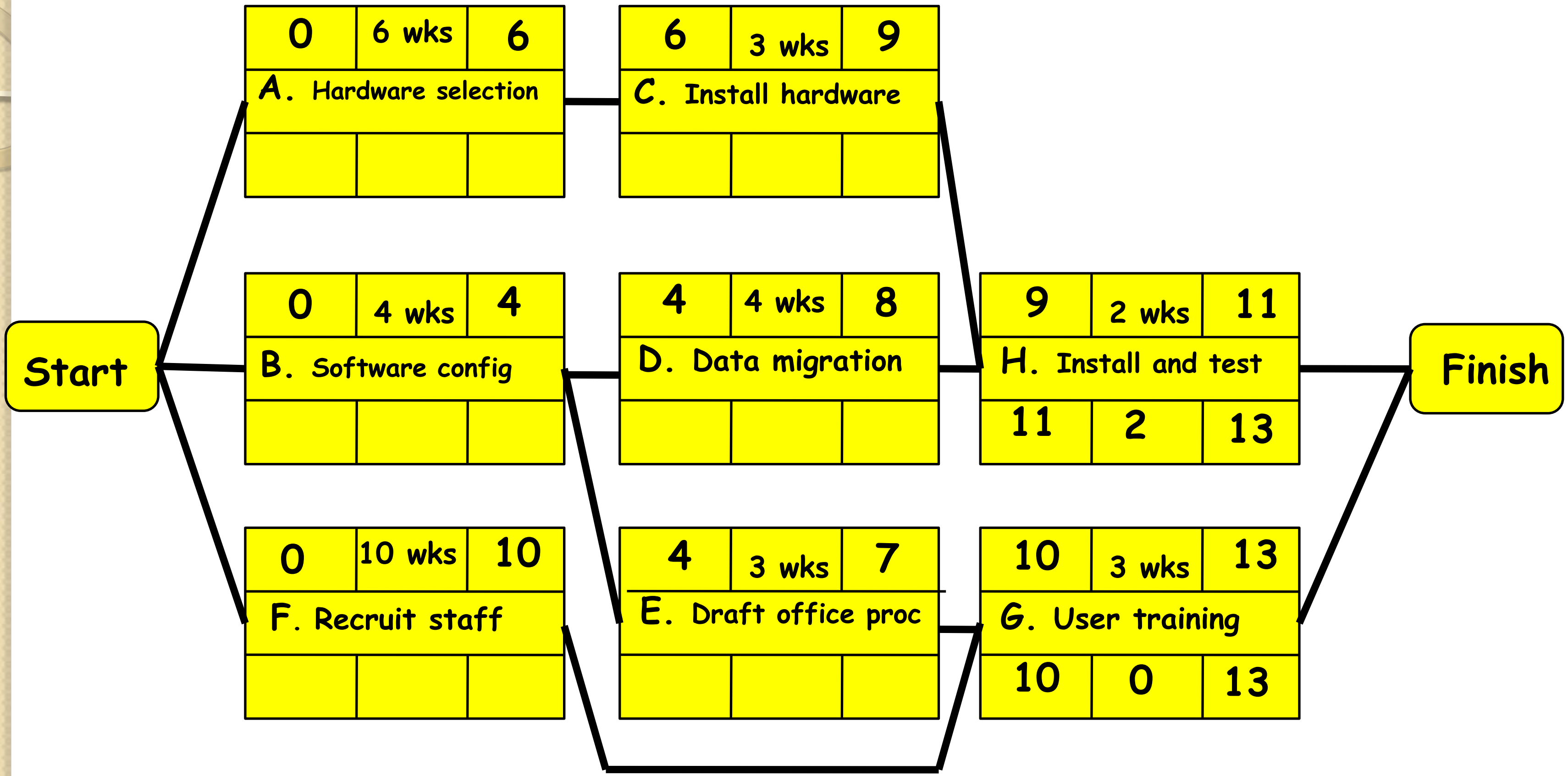
After Forward Pass Step 3 (Final)



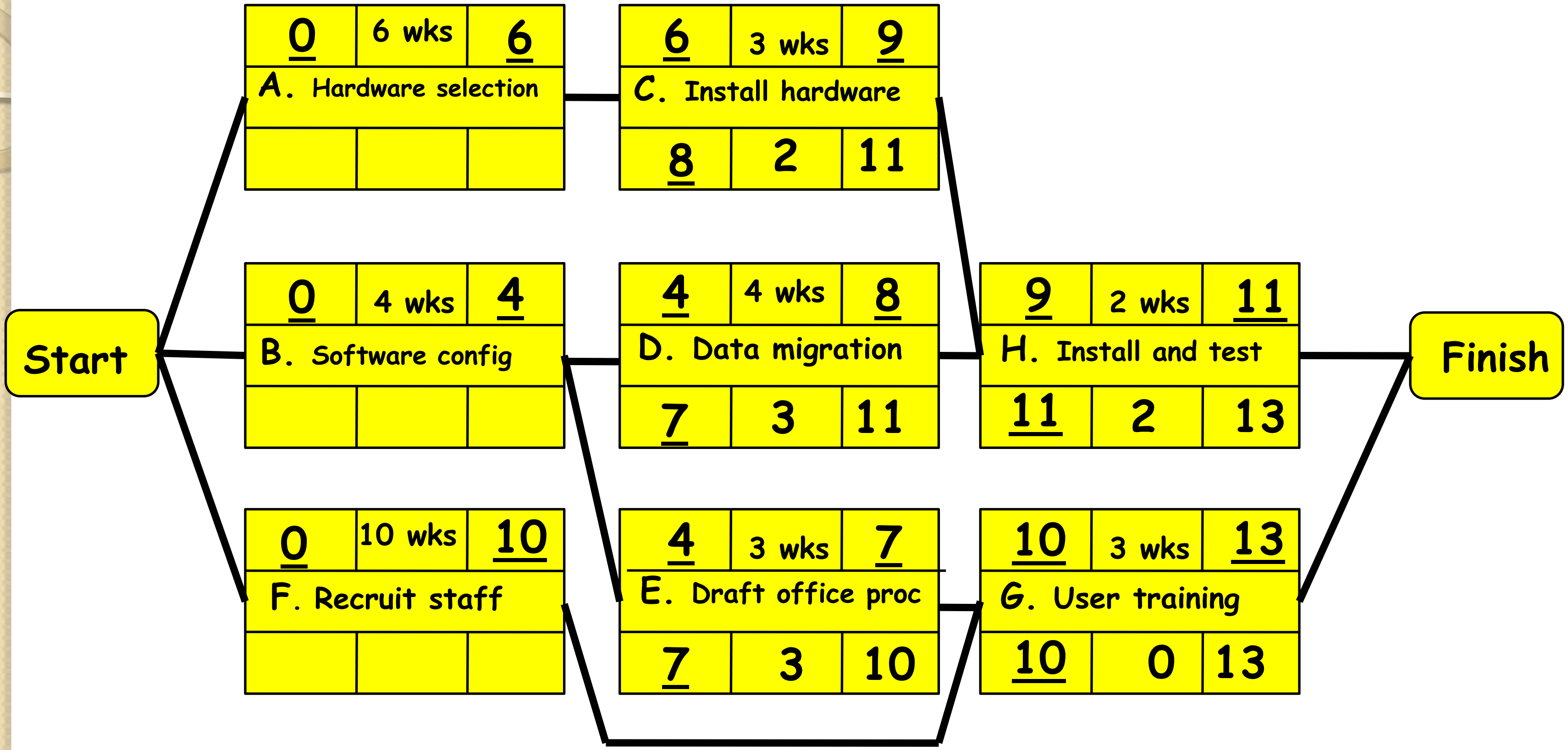
The Backward Pass

- The backward pass is carried out to calculate the latest start date at which each activity may started and finished without delaying the end date of the project.
- Rule
  - The latest finish date for an activity is the latest start date for the activity that commences immediately that activity is complete.
  - Where more than one activity can commence we take the earliest of the latest start dates for those activities.

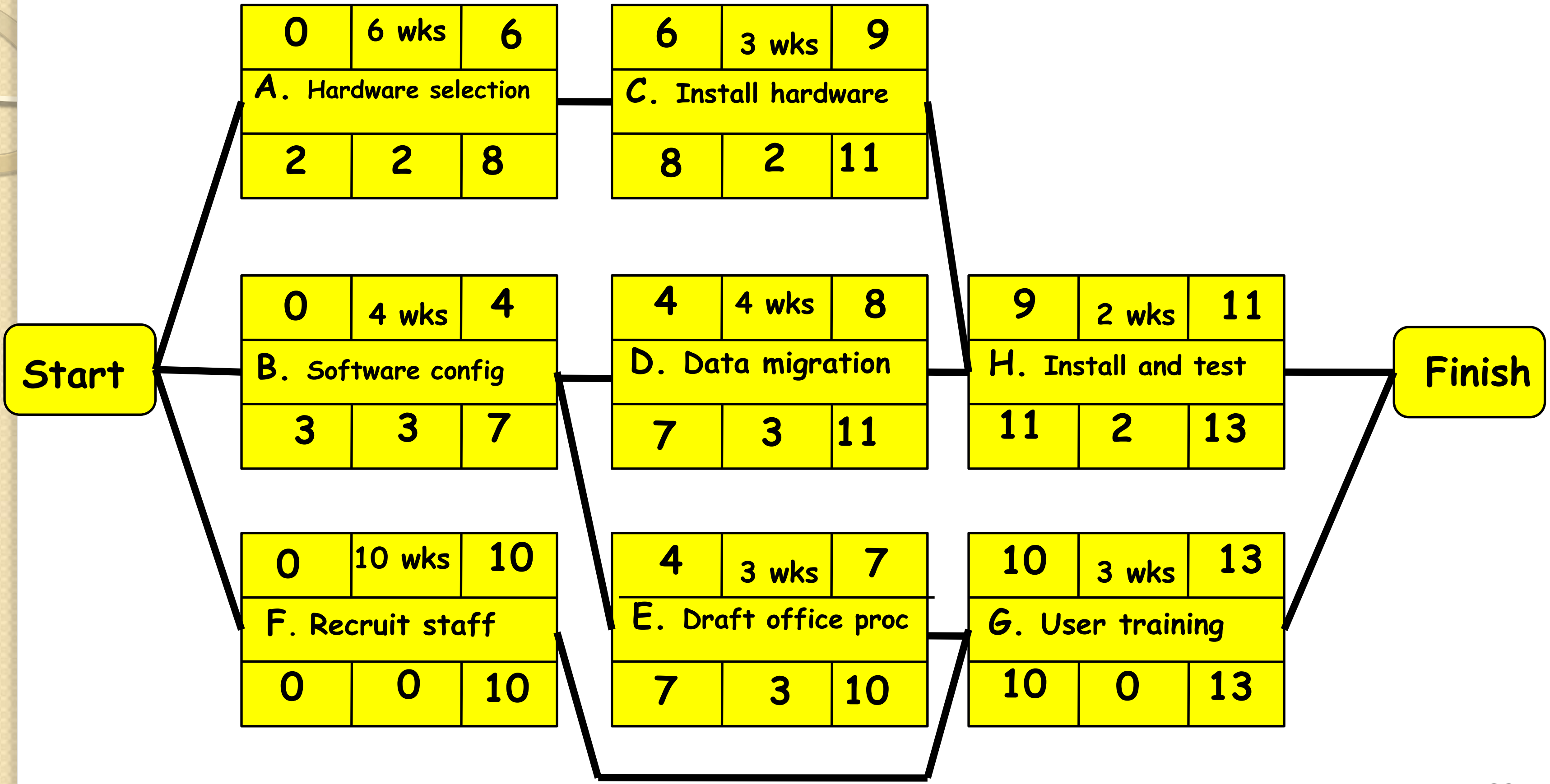
After Backward Pass Step 1



After Backward Pass Step 2



After Backward Pass Step 3 (Final)

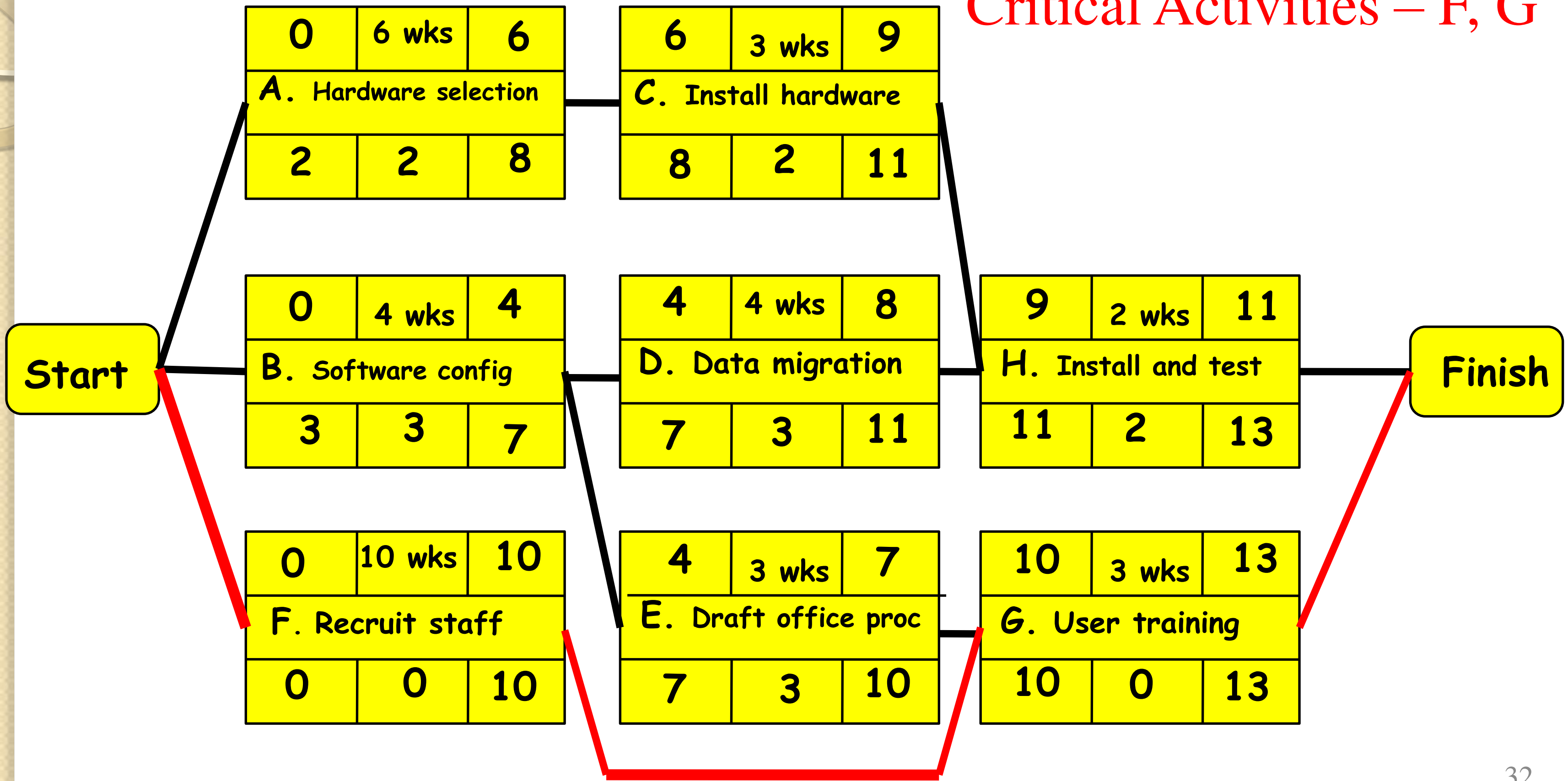


Critical Path

- The critical path is the longest path through the network.
- The difference between an activity's earliest start date and its latest start date (or equally, the difference between its earliest and latest finish dates) is known as activity's *float*.
- Any activity with a float of zero is critical in the sense that
  - Any delay in carrying out the activity will delay the completion date of the project as a whole.

Critical Path

Critical Activities – F, G



Activity float

- Free float
  - The time by which an activity may be delayed without affecting any subsequent activity.
  - It is calculated as the difference between the earliest completion date for the activity and the earliest start date of the succeeding activity.
- Interfering float
  - The difference between total float and free float.
  - It tells by how much the activity may be delayed without delaying the project end date
    - Even though it will delay the start of subsequent activities.

Shortening the Project Duration

- If we wish to shorten the overall duration of a project we would normally consider attempting to reduce activity durations. In many cases this can be done by applying more resources to the task - working overtime or procuring additional staff, for example.
- The critical path indicates where we must look to save time - if we are trying to bring forward the end date of the project, there is clearly no point in attempting to shorten non-critical activities. Referring to next figure, it can be seen that we could complete the project in week 12 by reducing the duration of activity F by one week (to 9 weeks).

Shortening the Project Duration cont…

- As we reduce activity times along the critical path we must continually check for any non critical path emerging and redirect our attention where necessary There will come a point when we can no longer safely, or cost-effectively, reduce critical activity durations in an attempt to bring forward the project end dale.
- Further savings, if needed, must be sought in a consideration of our work methods and by questioning the logical sequencing of he Generally, time savings are to be found by increasing the amount of parallelism in the network and the removal of bottlenecks (subject always, of course, to resource and quality constraints).

Summary

- Discussed to rules to construct activity networks
- Solved another example on finding the critical activities and critical path
- Discussed how to shorten the project duration

References:

1. B. Hughes, M. Cotterell, R. Mall, *Software Project Management*, Sixth Edition, McGraw Hill Education (India) Pvt. Ltd., 2018.
2. R. Mall, *Fundamentals of Software Engineering*, Fifth Edition, PHI Learning Pvt. Ltd., 2018.

Thank you