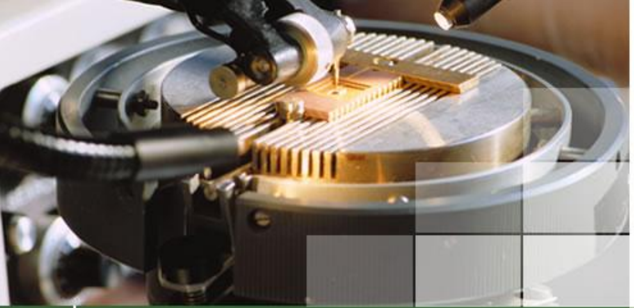
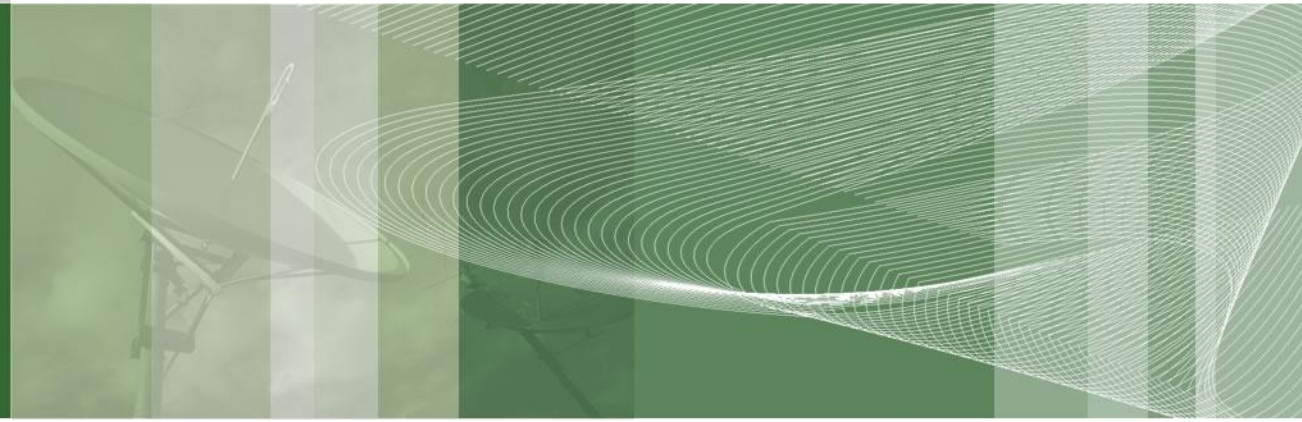


POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione



Planning and Managing Software Projects 2011-12
Class 9

Scheduling
Fundamentals, Techniques, Optimization

Emanuele Della Valle, Lecturer: Dario Cerizza
http://emanueledellavalle.org

Credits

- This slides are partially based on CEFRIEL's slides for PMI Certification and largely based on Prof. John Musser class notes on "Principles of Software Project Management"
- Original slides are available at http://www.projectreference.com/
- Reuse and republish permission was granted

Today

- **Class 7 and 8 Review**
- Scheduling Fundamentals
- Scheduling Techniques
  - Network Diagrams
  - Bar Charts
- Schedule Optimization Techniques

Class 7 and 8 Review

- WBS
- Estimation

WBS

- Types: Process, Product, Hybrid
- Formats: Outline or graphical organizational chart
- High-level WBS does not show durations and dependencies
- WBS becomes input to many things, esp. schedule
- What hurts most is what's missing → 100% Rule:
  - the sum of the work on the children must be equal to 100% of the work referred by the parent
  - no work outside the scope of the project is in the WBS

Estimation

- Estimation is the process of determining the effort and the duration of activities

- Setting realistic expectations is the single most important task of a project
- "Unrealistic expectations based on inaccurate estimates are the single largest cause of software failure."
  - Futrell, Shafer, Shafer, "Quality Software Project Management"

Estimation

- Use multiple methods if possible
  - This reduces your risk
  - If using “experts”, use two
- History is your best ally
  - Especially when using Function Points, LOC (Lines of Code), …
- Get buy-in
  - Involve who will do the work in the estimation process
- Remember: estimation is an iterative process!
  - Esteems must be updated during project execution
- Know your “presentation” techniques
  - Esteems ranges
  - Different esteems with different probabilities

POLITECNICO DI MILANO Dipartimento di Elettronica e Informazione

Estimation

- Bottom-up
  - More work to create but more accurate
  - Often with Expert Judgment at the task level (buy-in)
- Top-down
  - Used in the earliest phases
  - Usually as is the case with Analogy or Expert Judgment
- Analogy
  - Comparison with previous project: formal or informal
- Expert Judgment
  - Via staff members who will do the work
  - Most common technique along with analogy
  - Best if multiple ‘experts’ are consulted

Estimation

- Parametric Methods
  - Know the trade-offs of: LOC & Function Points

- Function Points
  - Benefit: relatively independent of the technology used to develop the system
  - We will re-visit this briefly later in semester (when discussing “software metrics”)

Today

- **Scheduling Fundamentals**
- Scheduling Techniques
  - Network Diagrams
  - Bar Charts
- Schedule Optimization Techniques

Initial Planning, Estimating and Scheduling

- Initial Planning:
  - It’s needed for the SOW and the Project Charter
  - It answers to the question: What/How
    - WBS (1st pass)
    - Should consider activities for other project management documents
      - Software Development Plan, Risk Mgmt., Cfg. Mgmt.
- Estimating
  - It answers to the question: How much/How long
    - Size (quantity/complexity) and Effort (duration)
    - It’s an iterative process
- Scheduling
  - It answers to the question: In which order
    - Precedence, concurrences, lag & lead times, slack & float, …
    - It’s an iterative process

What Scheduling is

- Once tasks (from the WBS) and effort/duration (from estimation) are known: then “scheduling”
- Scheduling is the definition of the start and the end time of each activity/task
  - Requires the definition of **dependencies** among tasks and the **assignment of resources**
- The result of scheduling is a plan that defint:
  - **Who**
  - does **What**
  - by **When**

Scheduling Objectives

- Scheduling aims to find a trade-off among six objectives:
  - Primary objectives
    1. Best time
    2. Least cost
    3. Least risk
  - Secondary objectives
    4. Propose and evaluate different schedule alternatives
    5. Make an effective use of resources
    6. Reduce communication overhead among resources

POLITECNICO DI MILANO Dipartimento di Elettronica e Informazione

4 reasons for dependencies among tasks

1. Mandatory Dependencies
   - These are “Hard logic” dependencies
   - Nature of the work dictates an ordering
   - Ex: UI design precedes UI implementation
   - Ex: Coding has to precede testing (*)
2. Discretionary Dependencies
   - “Soft logic” dependencies
   - Determined by the project management team
   - Process-driven
   - Ex: Discretionary order of creating certain modules
   - Ex: First Front-end, then Back-End (because the opposite order is also meaningful)

- (*) NOTE: substantial process innovation often take place when “hard logic” dependencies are shown to be wrong
  - Ex: Test first approaches

4 reasons for dependencies among tasks

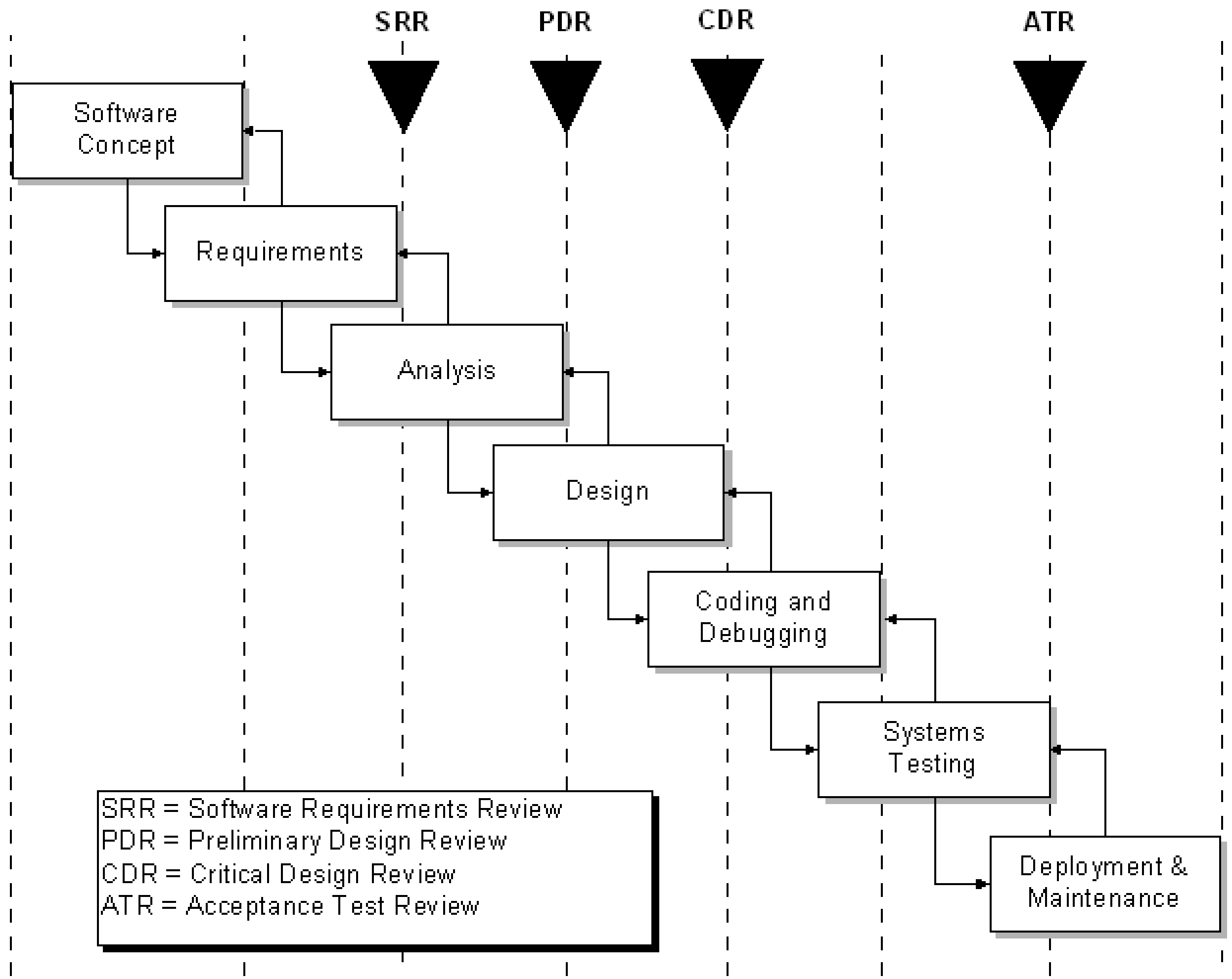
3. External Dependencies
   - Outside of the project itself
   - Ex: Release of 3rd party product; contract signoff
   - Ex: stakeholders, suppliers, year end

4. Resource Dependencies
   - Two task rely on the same resource
   - Ex: You have only one DBA but multiple DB tasks
   - Ex: You have only one server but different software versions needs multiple installations

Milestones

- Identify crucial points in your schedule
- Have a duration of zero
- Typically shown as inverted triangle or a diamond
- Often used at “review” or “delivery” times
  - Or at end or beginning of phases
  - Ex: Software Requirements Review (SRR)
  - Ex: Delivery of working component
- Can be tied to contract terms
  - Ex: User Sign-off

Example Milestones over the waterfall process



Today

- Scheduling Fundamentals
- **Scheduling Techniques**
  - Network Diagrams
  - Bar Charts
- Schedule Optimization Techniques

Scheduling Techniques

- **Network Diagrams**
  - CPM
  - PERT
- Bar Charts
  - Milestone Chart
  - Gantt Chart

Network Diagrams

- Developed in the 1950’s
- A graphical representation of the tasks necessary to complete a project
- Clearly visualizes the flow of tasks & relationships

CPM & PERT

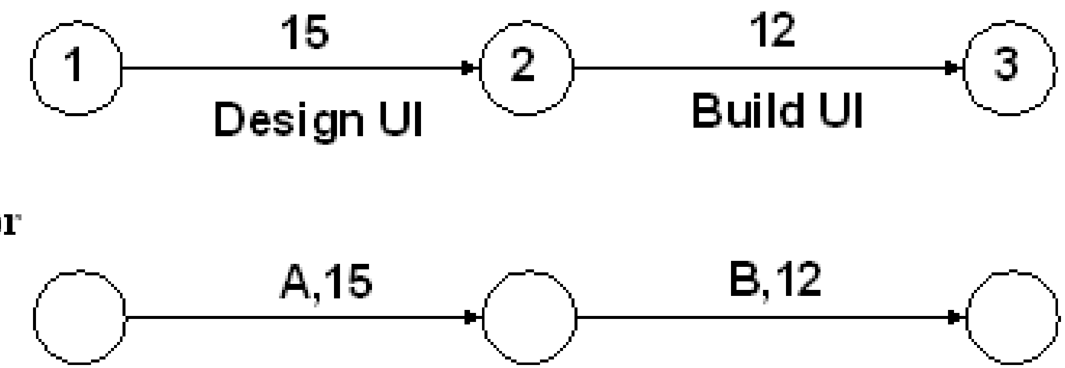
- CPM
  - Critical Path Method
- PERT
  - Program Evaluation and Review Technique
- Sometimes treated synonymously
- All are models using network diagrams

Two classic formats

- Two classic formats
  - AOA: Activity on Arrow
  - AON: Activity on Node
- Each activity labeled with
  - Identifier (usually a letter/code)
  - Duration (in standard unit like days)
- There are other variations of labeling
- There is one start & one end event
- Time goes from left to right

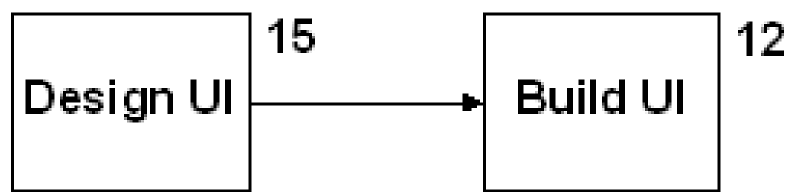
Formats

Activity on Arrow (AOA)

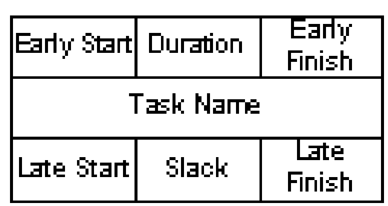


or

Activity on Node (AON)



or

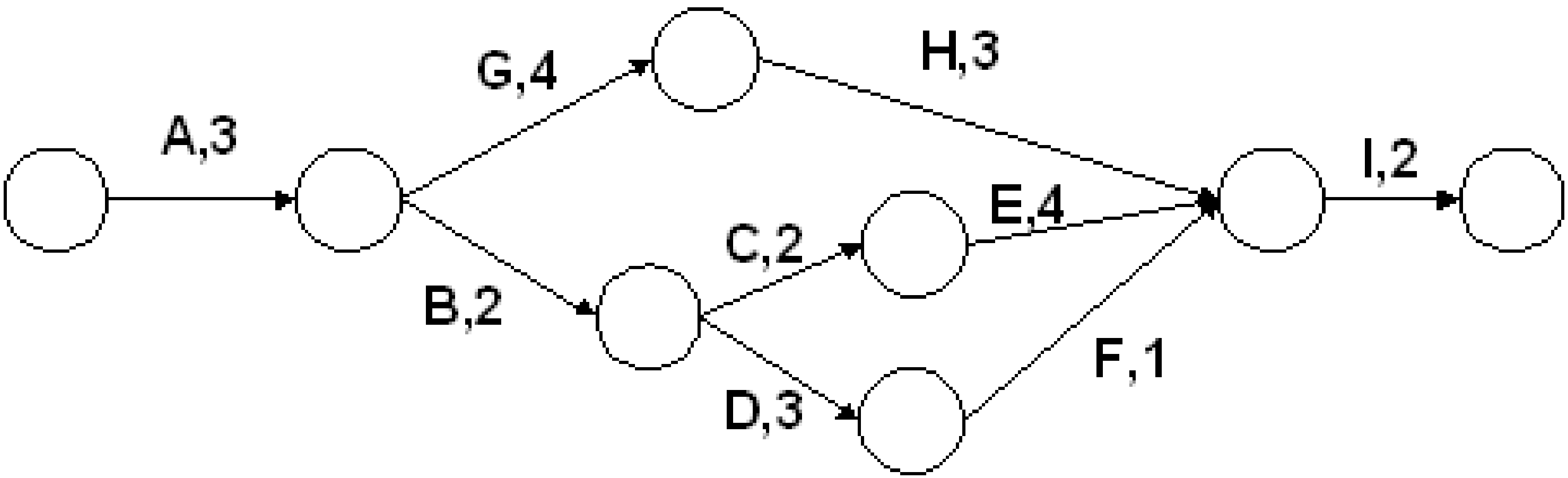


Formats

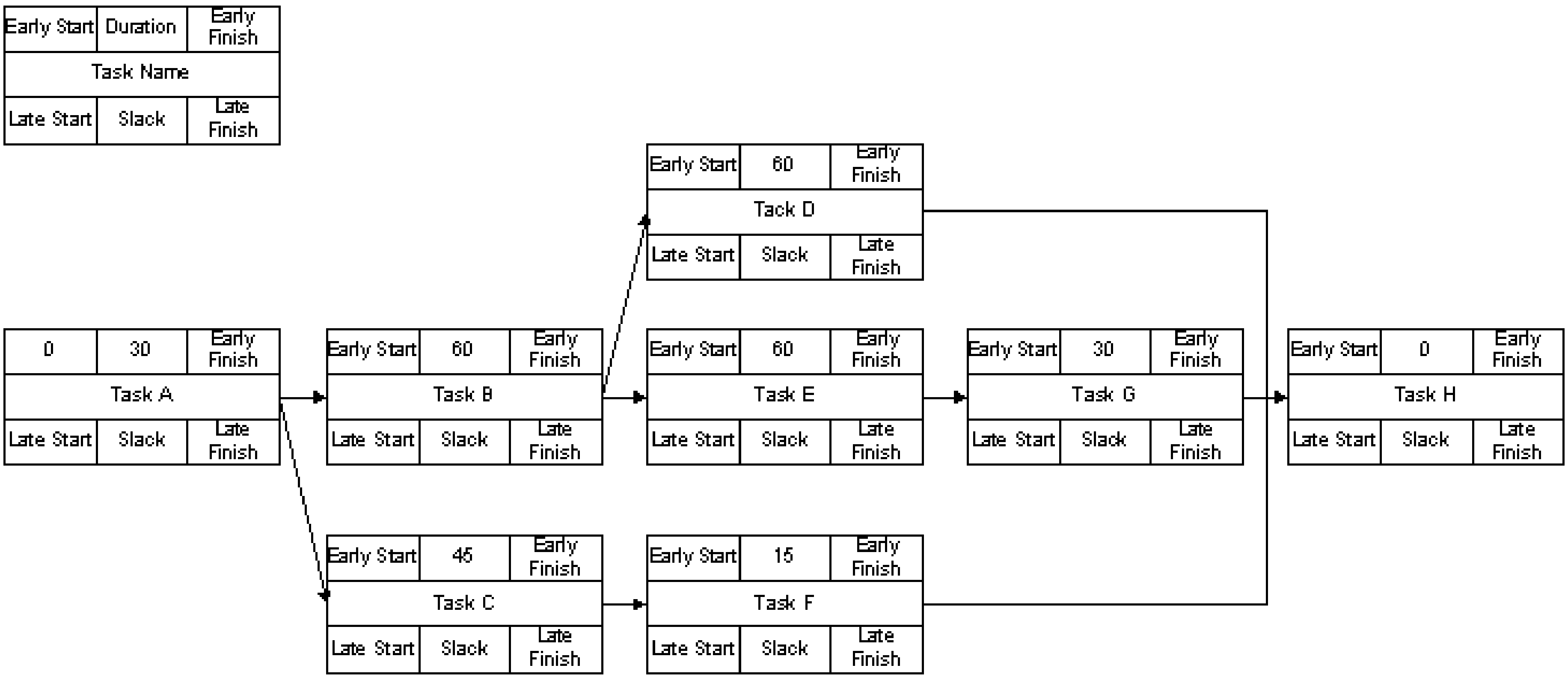
- AOA
  - Activities on Arrows
  - Circles representing Events
    - Such as ‘start’ or ‘end’ of a given task
  - Lines representing Tasks
    - Thing being done ‘Build UI’
  - a.k.a. Arrow Diagramming Method (ADM)

- AON
  - Activities on Nodes
    - Nodes can be circles or rectangles (usually latter)
    - Task information written on node
  - Arrows are dependencies between tasks
  - a.k.a. Precedence Diagramming Method (PDM)

AOA Example



AON Example

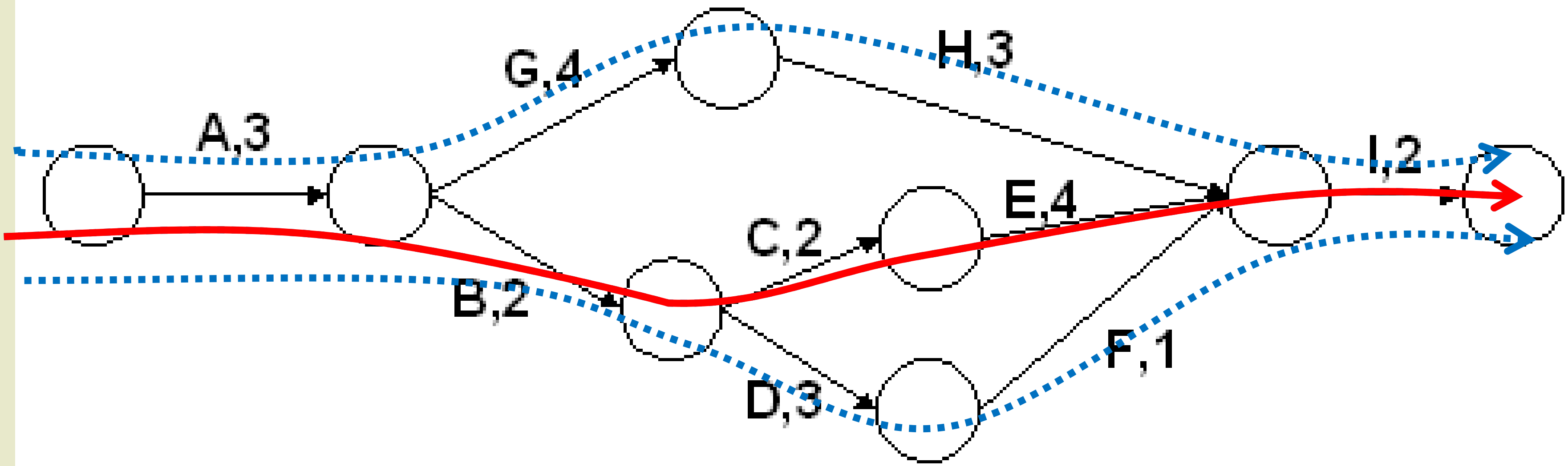

Early Start
Duration
Early Finish
Task Name
Late Start
Slack
Late Finish
Early Start
60
Early Finish
Tack D
Late Start
Slack
Late Finish
0
30
Early Finish
Task A
Late Start
Slack
Late Finish
Early Start
60
Early Finish
Task B
Late Start
Slack
Late Finish
Early Start
60
Early Finish
Task E
Late Start
Slack
Late Finish
Early Start
30
Early Finish
Task G
Late Start
Slack
Late Finish
Early Start
0
Early Finish
Task H
Late Start
Slack
Late Finish
Early Start
45
Early Finish
Task C
Late Start
Slack
Late Finish
Early Start
15
Early Finish
Task F
Late Start
Slack
Late Finish

Critical Path

- A Critical Path is a specific set of sequential tasks upon which the project completion date depends
- Tasks on the critical path cannot be delayed without delaying the project completion day
  - If a task on the critical path is delayed by 1 day, then the project completion date is delayed (at least) by 1 day
- All projects have at least one Critical Path
  - Critical Paths are the paths with duration = total project duration

Critical Path: an Example on AOA diagram

- How many paths are here?
- Which one is the Critical Path?



Critical Path: some Corollaries

- Delaying tasks on the critical path directly lengthens the schedule
- Accelerating tasks on the critical path does NOT directly shorten the schedule
  - Critical Path may change to another as you shorten the current critical path
- Delaying tasks NOT on the critical path does NOT directly lengthen the schedule
  - Tasks out of the critical path have a little amount of time (named **slack**) that let them to be delayed without impacting on the project completion date
- Accelerating tasks NOT on the critical path does NOT change the project completion date
  - Their slack increases

Critical Path Method (CPM)

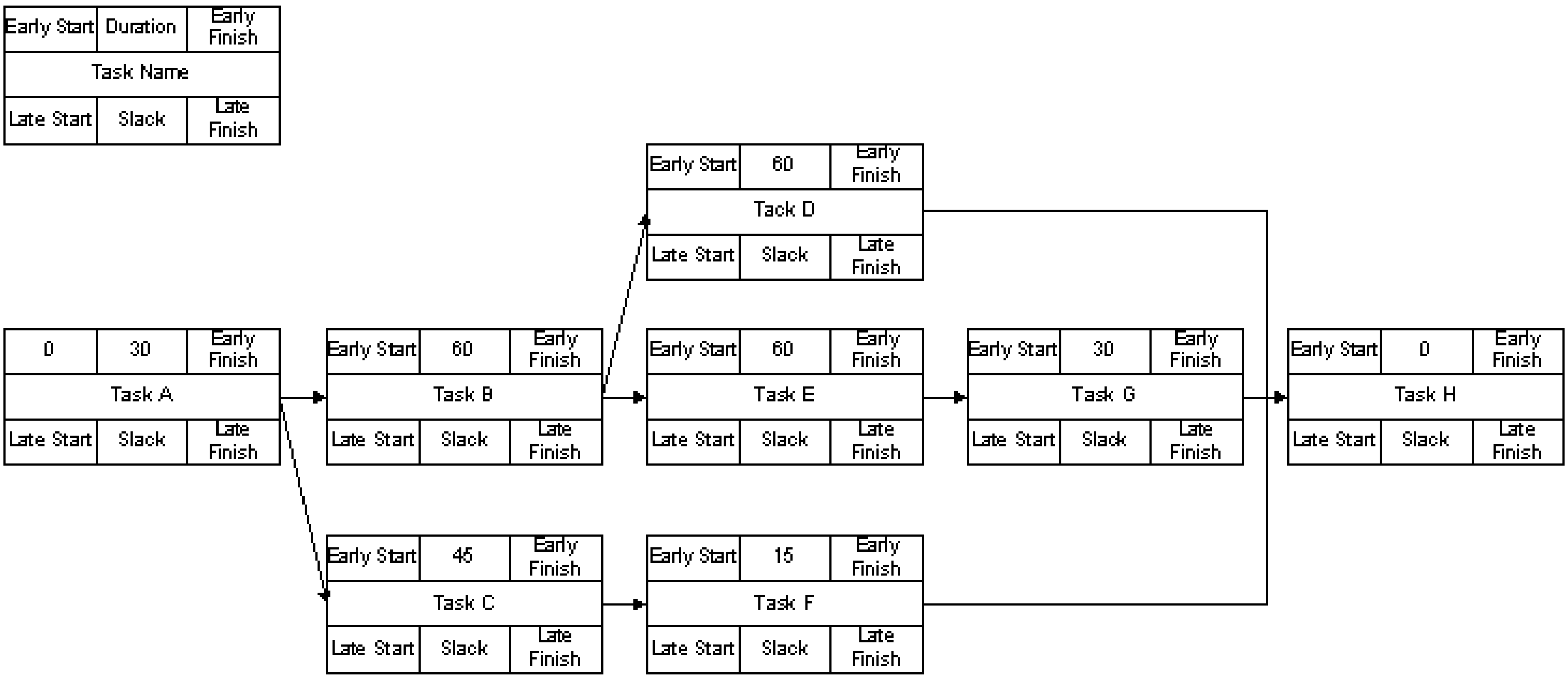
- The process for determining and optimizing the critical path
- Should be done in conjunction with the project manager & the functional manager
- Based upon a **2-passes approach**
  - **Forward Pass** and **Backward Pass**
- As result of the 2-passes, the critical path becomes evident

(This exercise is part of course exams!)

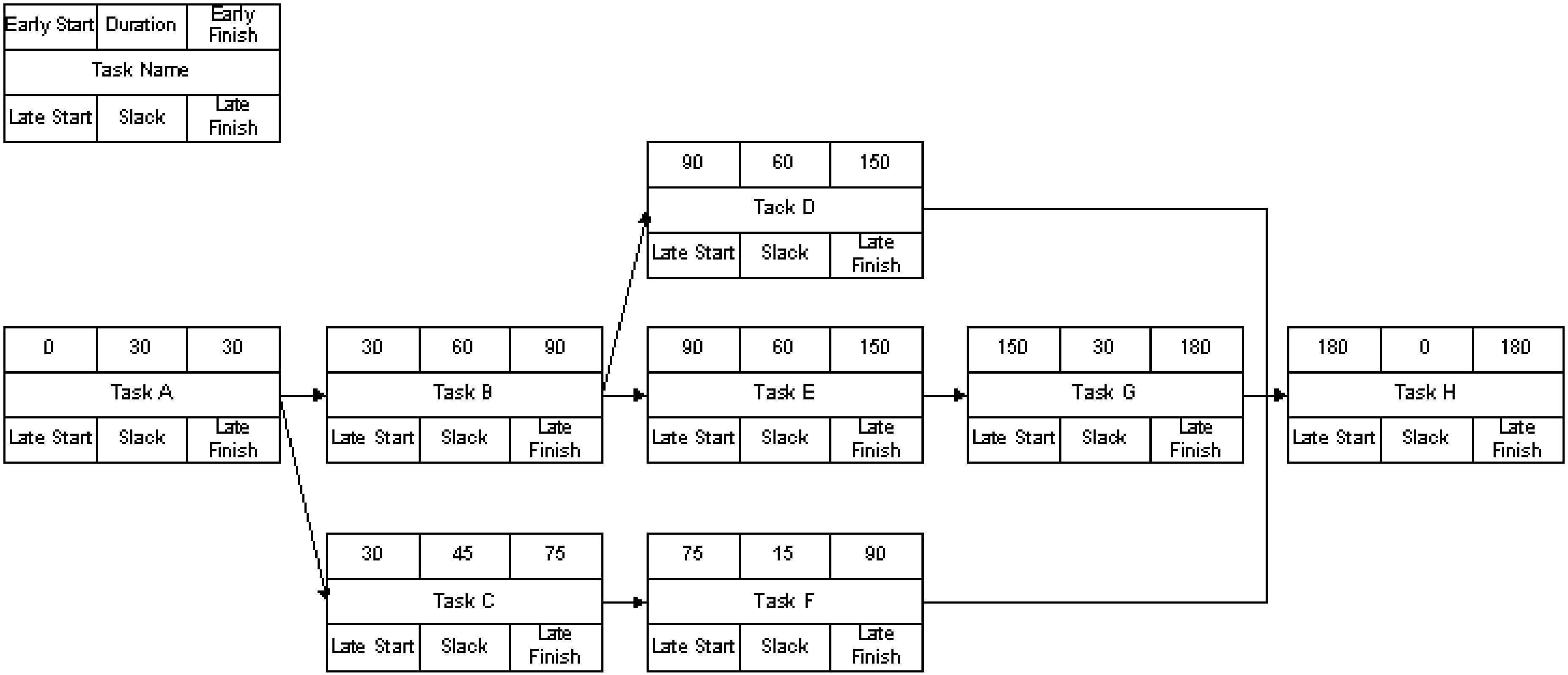
CPM: Forward Pass

- Used to determine early start (ES) and early finish (EF) times for each task
- Work from left to right
- Adding times to each node and each path
- Rule: when several tasks converge, the ES for the next task is the **largest** of preceding EF times

CPM: Example: Forward (part 1)



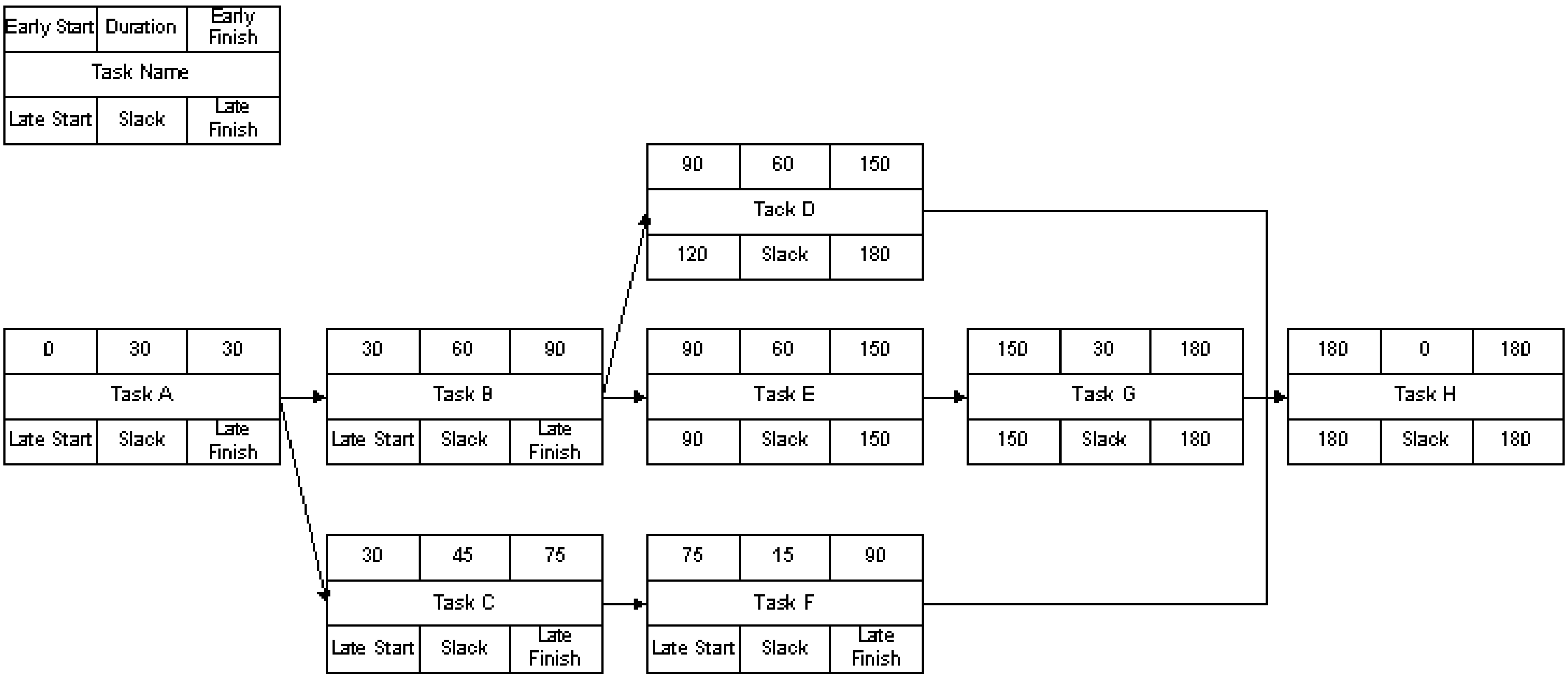
CPM: Example: Forward (part 2)



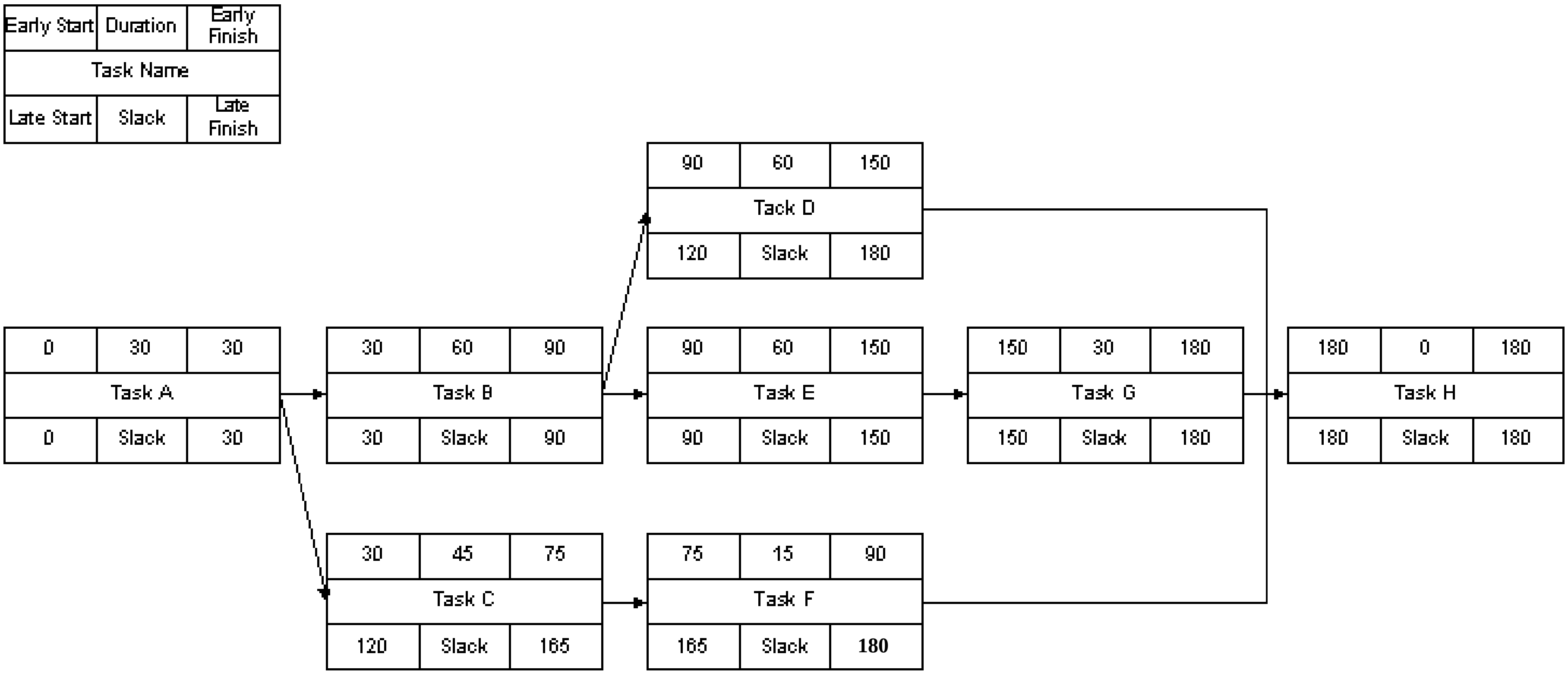
CPM: Backward Pass

- Used to determine the late finish (LF) and late start (LS) times
- Start at the end node and move backward left
- Subtract duration from connecting node's earliest start time
- Rule: when several tasks converge, the last finish for the previous task is the **smallest** of following last start times

CPM: Example: Backward (part 1)



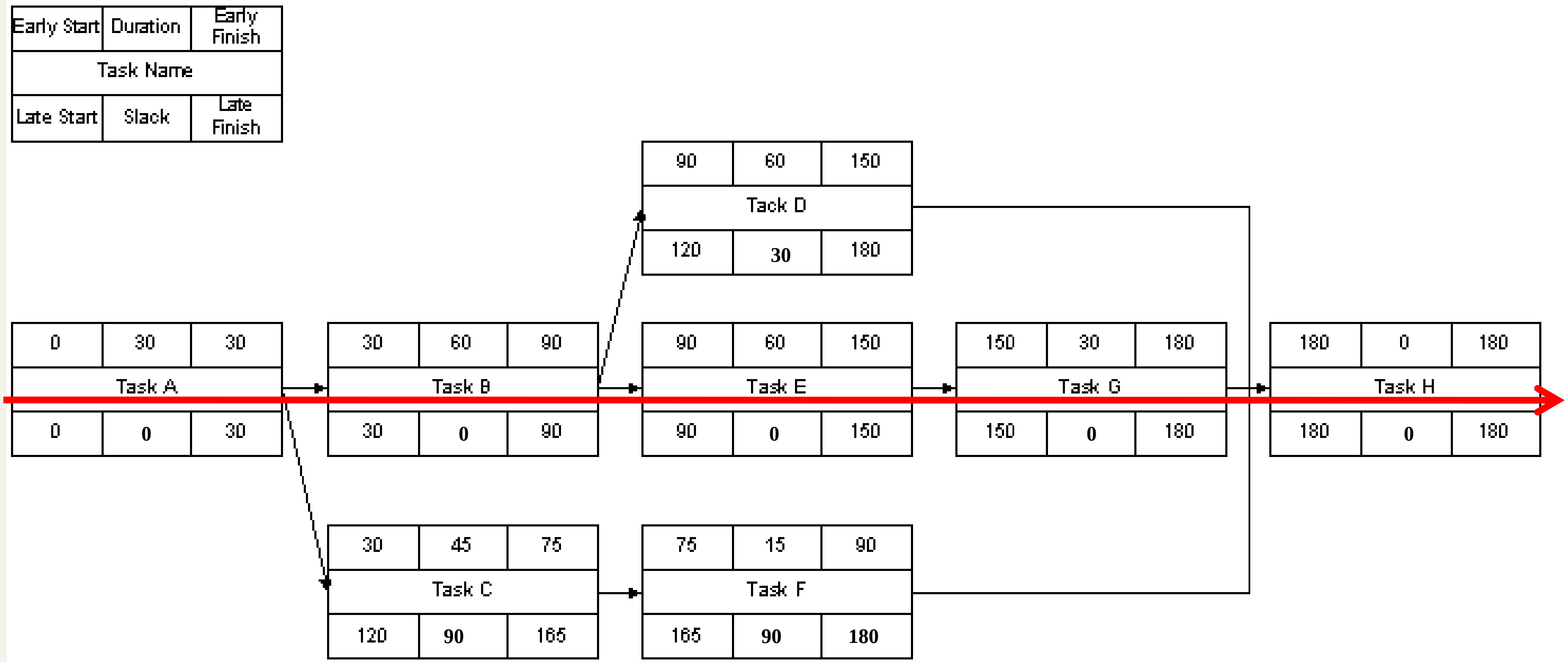
CPM: Example: Backward (part 2)



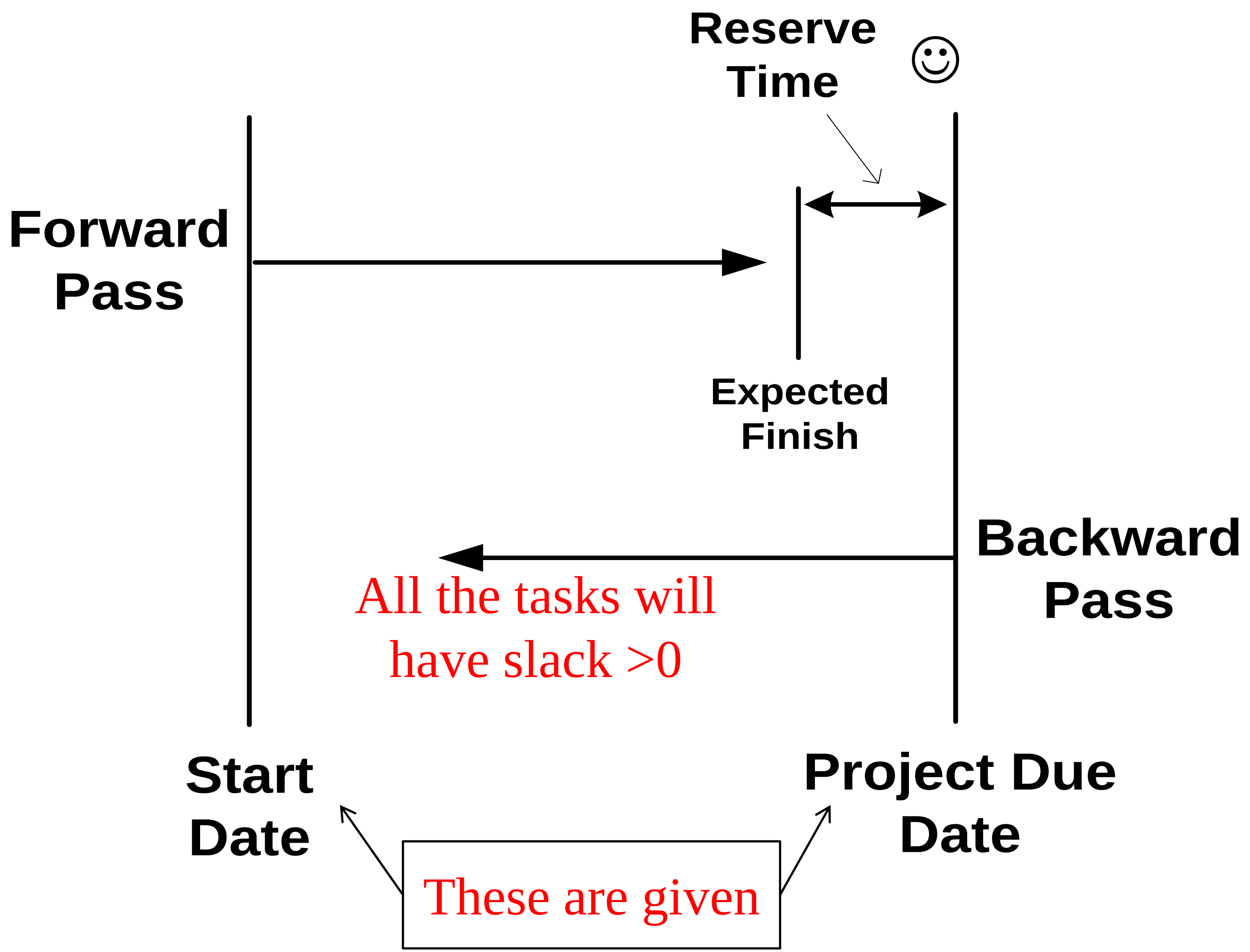
CPM: Example: Compute Slacks and Identify Critical Paths

Slack = Late Finish – Early Finish = Late Start – Early Start

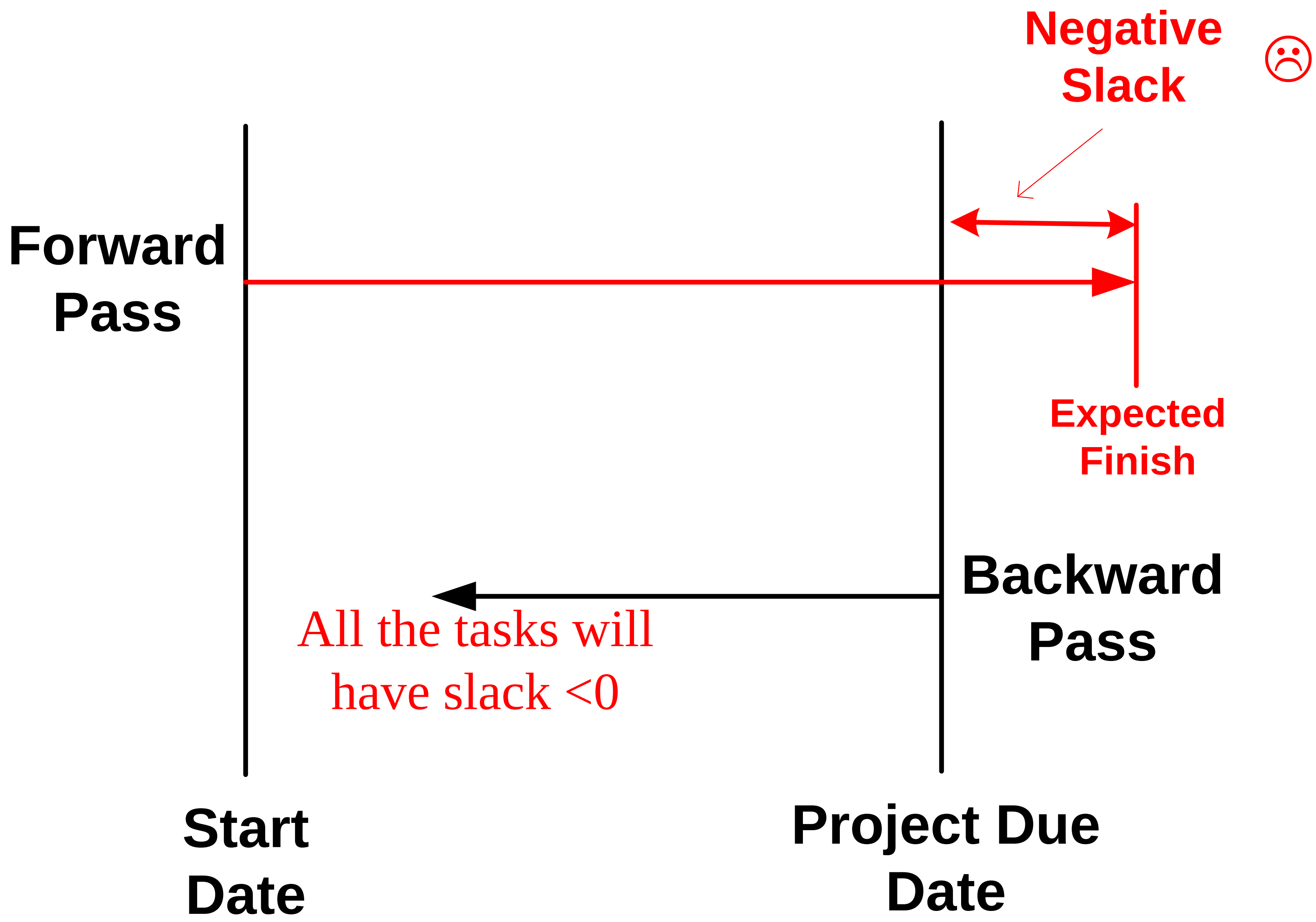
Critical Path: all tasks with slack = 0



CPM and Slack & Reserve



CPM and Slack & Reserve



CPM: Advantages and Disadvantages

- Advantages
  - Show precedence well
  - Reveal interdependencies not shown in other techniques
  - Ability to calculate critical path
  - Ability to perform “what if” exercises
- Disadvantages
  - Default model assumes resources are unlimited
    - You need to incorporate this yourself (Resource Dependencies) when determining the “real” Critical Path
  - Difficult to follow on large projects

PERT

- Program Evaluation and Review Technique
- Based on idea that estimates are uncertain
  - Therefore uses duration ranges
  - And the probability of falling to a given range
- First is done on each task, then at project level

For each task:

1) Start with 3 estimates for each task
   - Optimistic
     - Would likely occur 1 time in 20
   - Most likely
     - Modal value of the distribution
   - Pessimistic
     - Would be exceeded only one time in 20

PERT Formula

2) Calculate the expected time of each task:

$$t_e = \frac{a + 4m + b}{6}$$

where

t_e = expected time

a = optimitistic time estimate

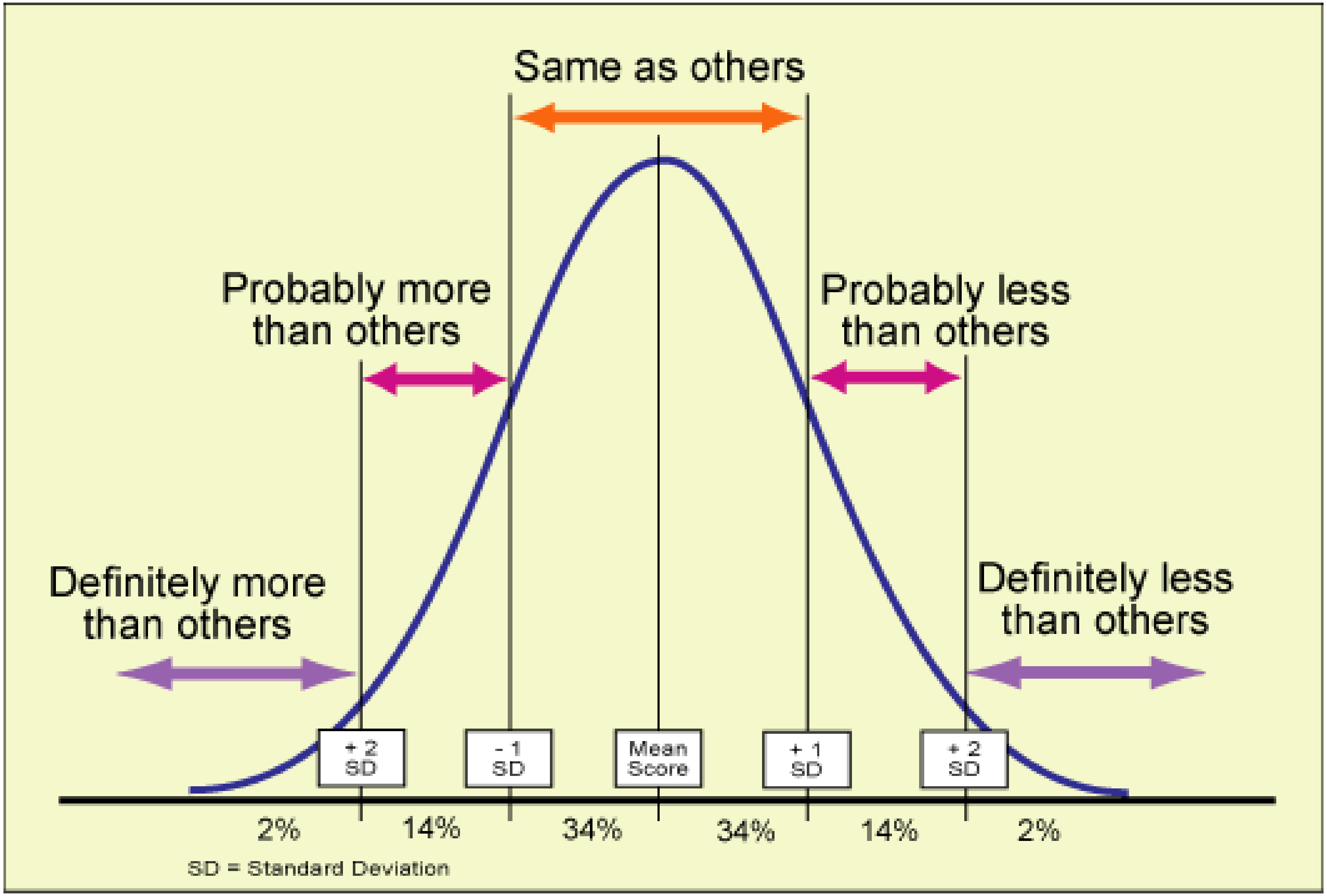
m = most likely time estimate

b = pessimistic time estimate

3) Calculate the standard deviation of each task:

$$s_i = \frac{b_i - a_i}{6}$$

MEMO: bell curve (normal distribution)



[source : http://www.fontys.nl/lerarenopleiding/tilburg/engels/Toetsing/bell_curve2.gif]

PERT Example on a single task

- Planner 1 (P1) and Planner 2 (P2) are asked to estimate m, a and b for a task

	Planner 1	**Planner 2**
m	10d	10d
a	9d	9d
b	12d	20d

- Calculate estimated time and standard deviation

	Planner 1	**Planner 2**
PERT time	10.2d	11.5d
Std. Dev.	0.5d	1.8d

- With the (34%+34%=) 68% of probability
  - Planner 1 says that task will last between 9.7 to 10.7 days
  - Planner 2 says that task will last between 9.7 to 13.3 days

PERT Formula

For the whole project:

1) Update the network diagram with the expected time of each task

2) Calculate the critical path with the CPM method

→ Result is the expected time of the whole project

3) For each task in the critical path, calculate the standard deviation of the project as:

$$s_{cp} = \sqrt{s_1^2 + s_2^2 + \ldots + s_n^2}$$

PERT: Advantages and Disadvantages

- Advantages
  - Accounts for uncertainty
- Disadvantages
  - Time and labor intensive
  - Assumption of unlimited resources is big issue
  - Lack of functional ownership of estimates
  - Mostly only used on large, complex project
- Get PERT software to calculate it for you

CPM vs. PERT

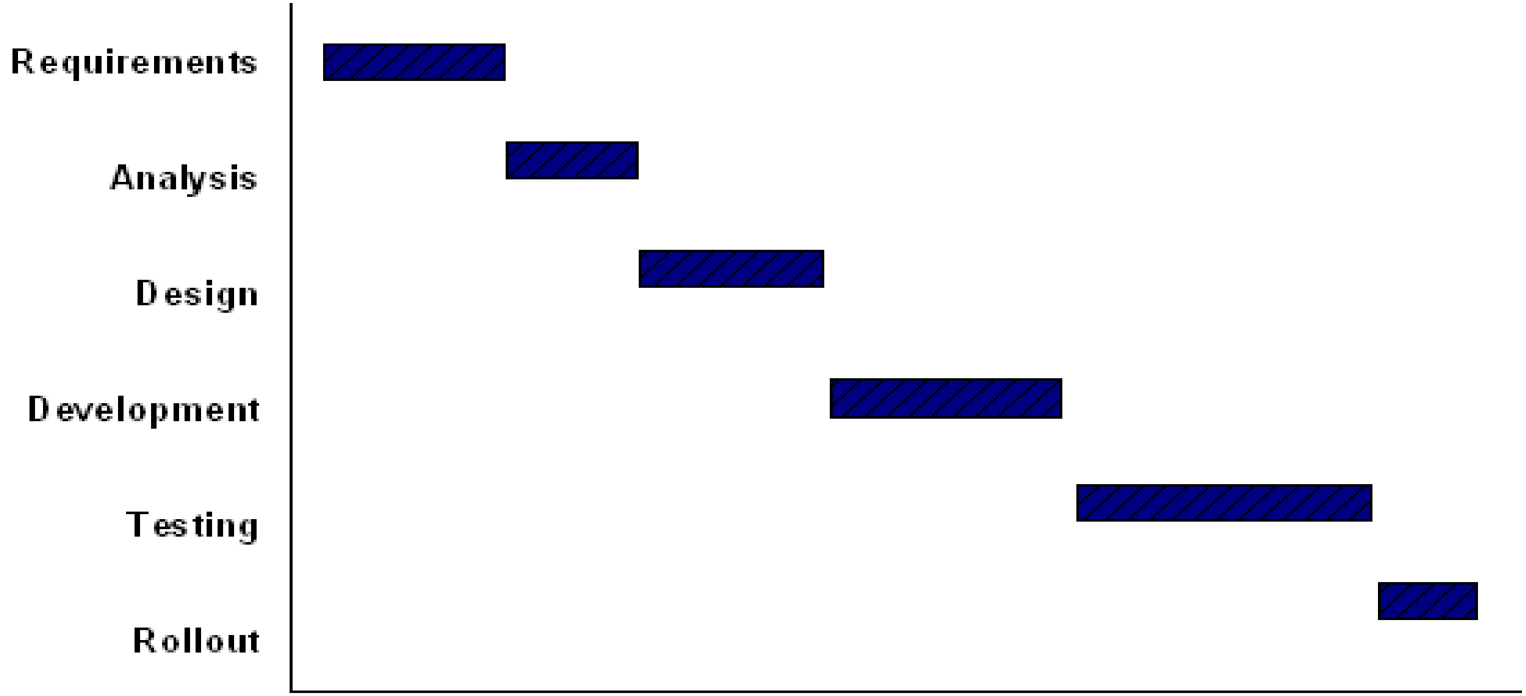
- Both use Network Diagrams
- CPM: deterministic
- PERT: probabilistic
- CPM: one estimate, PERT, three estimates
- PERT is infrequently used

Scheduling Techniques

- Network Diagrams
  - CPM
  - PERT
- **Bar Charts**
  - Milestone Chart
  - Gantt Chart

Milestone Chart

- It shows the temporal sequence of tasks
  - One bar per each task
  - Bar length is proportional to task duration
- It shows only more general tasks of the WBS



Gantt Chart

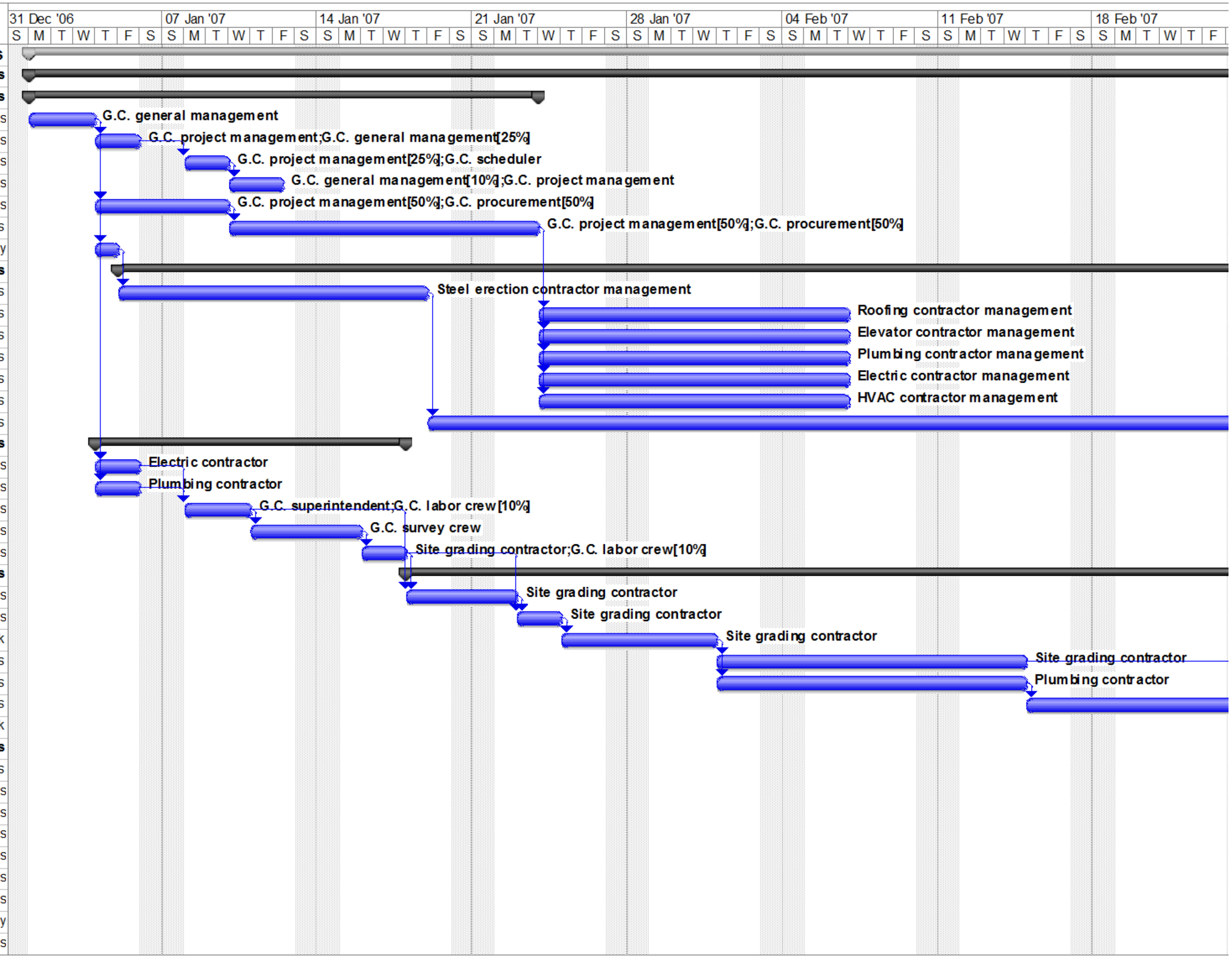
- As the milestone chart, it shows the temporal sequence of tasks and their duration
- Moreover
  - It consider all tasks organized in groups and subgroups
  - It shows dependencies among tasks
- It may also show:
  - Start and end dates of each task
  - Resources involved in each task
  - Percentage of completion of each task (useful during project controlling phases)
  - Today date

(Next lab lesson will entirely focus in drawing Gantt charts)

(A Gantt chart is required by last homework)

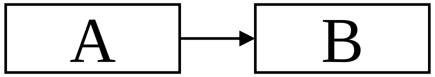
An example

ID		Task Name	Duration
0		Commercial Construction	355 days
1		Three-story Office Building (7	355 days
2		General Conditions	17 days
3		Receive notice to proc	3 days
4		Submit bond and insu	2 days
5		Prepare and submit pr	2 days
6		Prepare and submit sc	2,33 days
7		Obtain building permit:	4 days
8		Submit preliminary sh	2 wks
9		Submit monthly reque	1 day
10		Long Lead Procurement	70 days
11		Submit shop drawings	2 wks
12		Submit shop drawings	2 wks
13		Submit shop drawings	2 wks
14		Submit shop drawings	2 wks
15		Submit shop drawings	2 wks
16		Submit shop drawings	2 wks
17		Detail, fabricate, and c	12 wks
18		Mobilize on Site	10 days
19		Install temporary powe	2 days
20		Install temporary wate	2 days
21		Set up site office	3 days
22		Set line and grade ber	3 days
23		Prepare site - lay dowi	2 days
24		Site Grading and Utilitie	35 days
25		Clear and grub site	3 days
26		Stone site access anc	2 days
27		Rough grade site (cut	1 wk
28		Install storm drainage	2 wks
29		Install exterior fire line	2 wks
30		Perform final site gradi	2 wks
31		Erect building batter b	1 wk
32		Foundations	39 days
33		Excavate foundations	2 wks
34		Excavate elevator pit	2 days
35		Form column piers ani	4 days
36		Rough-in electric and		4 days
37		Form elevator pit walls	4 days
38		Set reinforcing and ani	4 days
39		Pour column piers anc	5 days
40		Pour concrete elevatoi	1 day
41		Cure elevator wall con	7 days

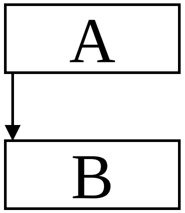


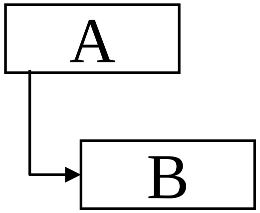
Precedence and Concurrence

- Precedence:
  - A task that must occur before another is said:
    - *To have **precedence** of the other*
      - or
    - *To be a **predecessor** of the other*



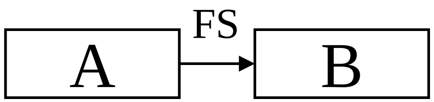
- Concurrence:
  - Concurrent tasks are those that can occur in parallel at the same time
    - Completely overlapped or Partially overlapped



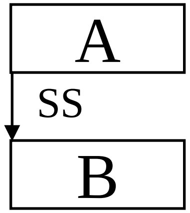


Dependency types

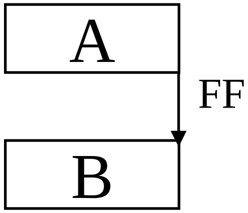
- *Finish-to-start*: “A f-to-s B”: B cannot start until A finishes. B can start after A finishes.



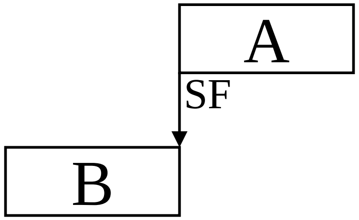
- *Start-to-start*: “A s-to-s B”: B cannot start until A starts. B can start after A starts.



- *Finish-to-finish*: “A f-to-f B“: B cannot finish until A finishes. B can finish after A finishes

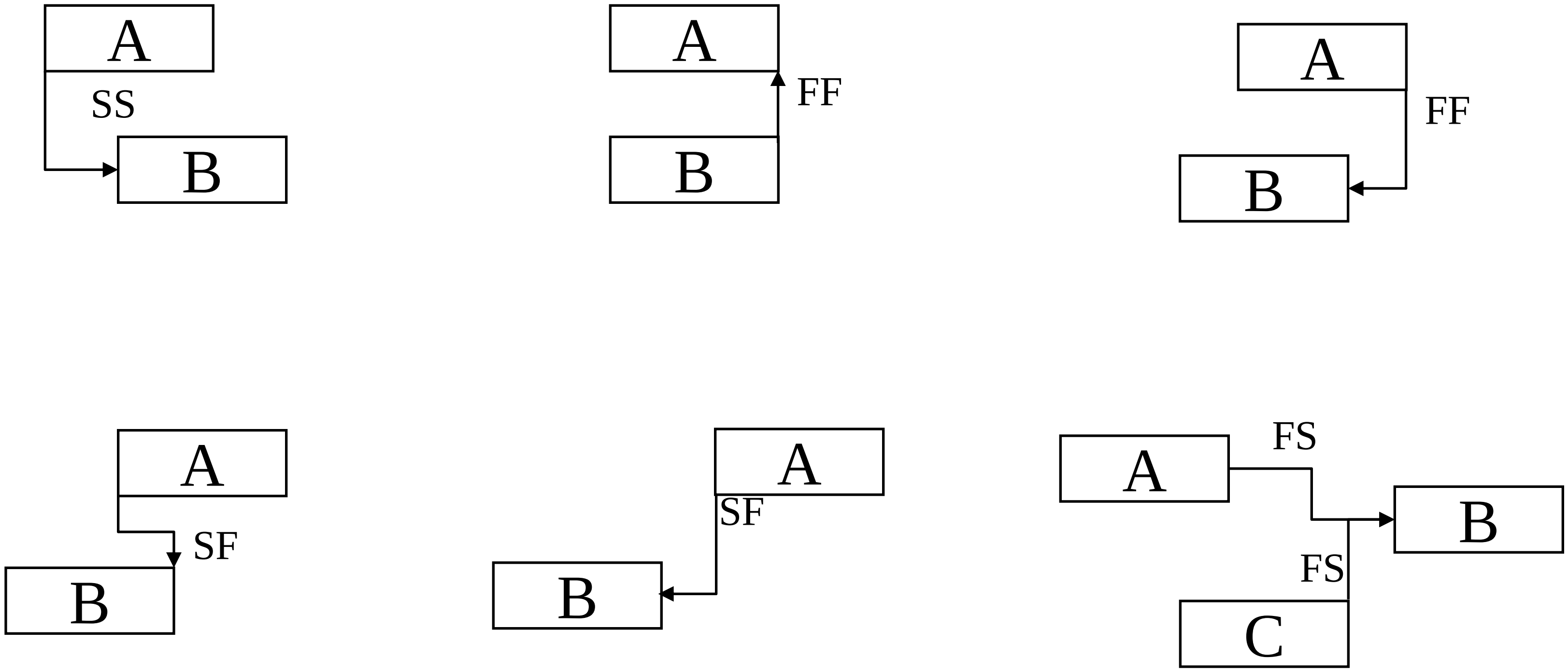


- *Start-to-finish*: “A s-to-f B“: B cannot finish until A starts. B can finish after A starts.



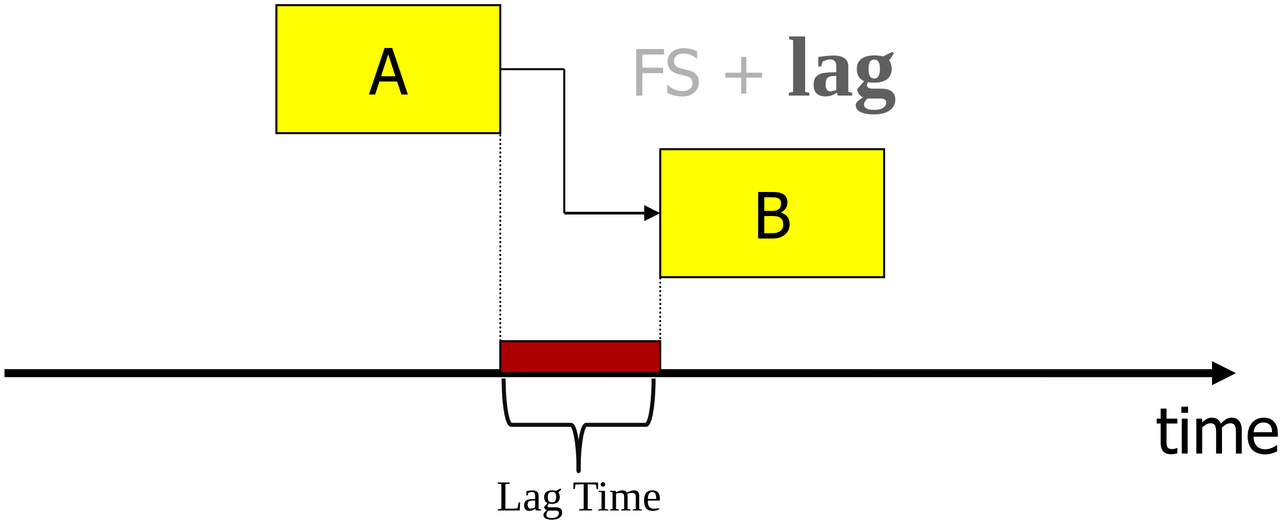
Dependency types (test)

- Which ones are wrong?



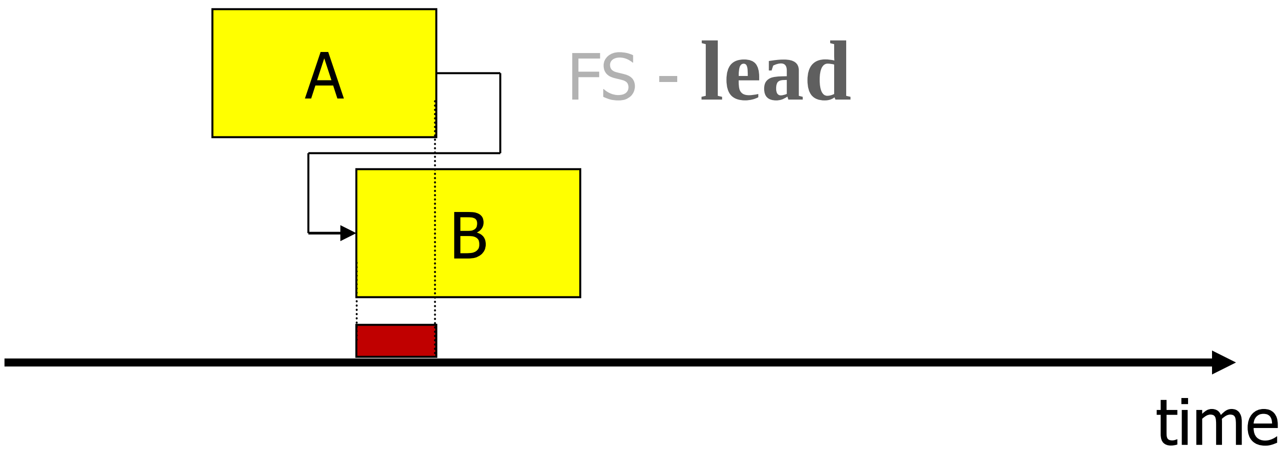
Lag Time

- It means a delay between tasks in sequence
- Example: if "A f-to-s B" and lag is equal to 10, B can start only 10 days after A end



Lead Time

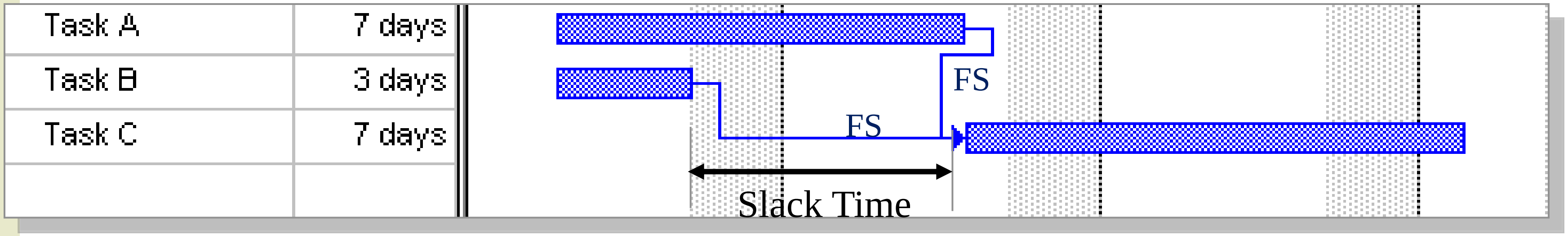
- It means an advance between tasks in sequence
- Example: if "A f-to-s B" and lead is equal to 10, B can start 10 days before A end



POLITECNICO DI MILANO
Dipartimento di Elettronica e Informazione

Slack & Float

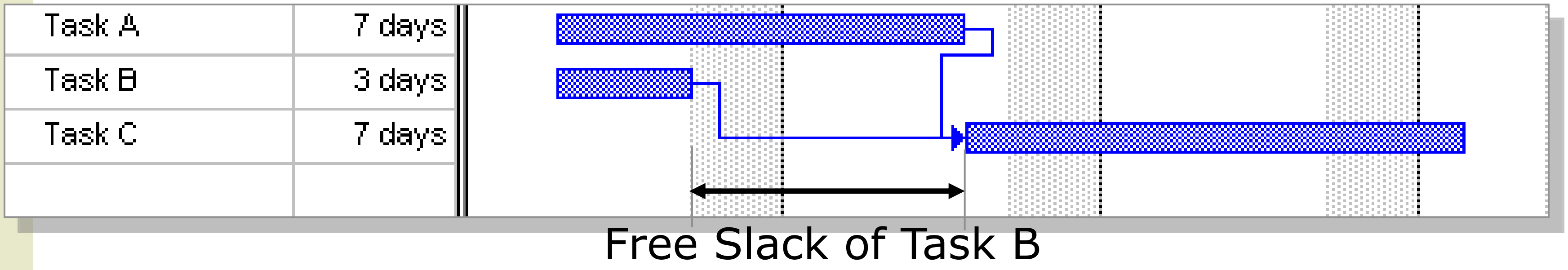
- When the schedule contains some concurrent tasks, it may happen that some “free” time is between tasks



- A **Slack** is the amount of time that a task can be delayed without causing a delay to:
  - any downstream task (named **Free Slack**)
  - the end of the project (named **Total Slack**)
- Slack & Float are synonymous terms
  - We’ll use Slack

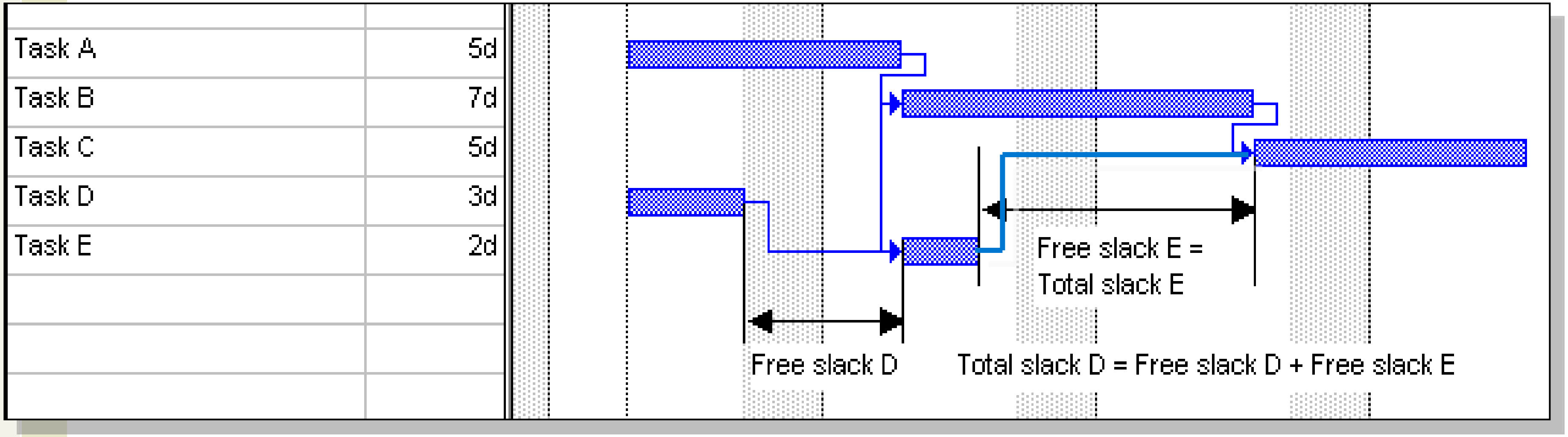
Free Slack

- Free Slack: Maximum delay of a task without causing a delay for any downstream task
- Example
  - A and B can be activated as soon as possible
  - Relationships
    - "A finish-to-start C"
    - "B finish-to-start C"



Total Slack

- Total Slack: Maximum delay of a task without causing a delay for the end of the project
  - It can cause delays to some downstream tasks



- Normally, if not specified, **slack** stands for **total slack** (as we did in the CPM)

Advantages and Disadvantages

- Advantages
  - Easily understood
  - Easily created and maintained
  - Good support from software tools

  → **Largely used**
- Disadvantages
  - It has difficulties to show complex relationships among tasks (network diagrams are better in this)
  - It does not show uncertainty of a given activity (as does PERT)

Today

- Scheduling Fundamentals
- Scheduling Techniques
  - Network Diagrams
  - Bar Charts
- **Schedule Optimization Techniques**

Schedule Optimization Techniques

- Several techniques may be adopted to shorten the schedule
  - They act on the Trade-off Triangle (time, scope, cost)

1. **Re-organize schedule**:
   - *Split and merge tasks* to reduce dependencies and slacks
   - *Move resources among tasks* to better spread their effort in the time
     - A senior developer may act as a junior admin if needed
   - Advantage: focus on a pure schedule optimization with no impact on scope and cost
   - Disadvantages: risks increase
     - The schedule may become difficult to respect
     - Delays have stronger impact on the project (since slacks are reduces)
     - Resources may loose their motivation

Schedule Optimization Techniques

2. **Reduce scope**:

- *Reduce requirements* and consequently remove whole tasks
  - Advantage: cost and time reduction
  - Disadvantage: customers may not accept this
  - e.g. less tests, worsen performances, less graphics

- *Reduce quality* of software doing some tasks faster
  - e.g. less tests, worsen performances, less graphics
  - Disadvantages:
    - Customers and marketers may not accept this
    - Poorly tested software may cost more in the long time due to dependencies with other parts

Schedule Optimization Techniques

3. **Increase cost**:

- *Crashing: Add resources to tasks*
  - To shorten tasks
    - E.g. 2 developers may do the same work in less time
  - To break resource dependencies
    - E.g. another developer permits to parallelize tasks that would otherwise be sequential
  - Disadvantage: increased costs
- *Fast Tracking: Break dependencies* (both discretionary and some mandatory dependencies)
  - Such tasks become overlapped even if they should be sequential
    - E.g. developing data model and business logic in parallel
  - Disadvantages:
    - New resources may be needed on concurrent tasks (cost increases)
    - Since some tasks need inputs from other parallel tasks, some effort is spent in working with unstable inputs and some rework may occur (cost increases)

Questions?