Part I 1: Cyclomatic complexity

Given the Algorithm quickSort of

```
void insertionSort(int[] array) {
  for (int i = 2; i < array.length; i++) {
    tmp = array[i];
    array[0] = tmp;
    int j = i;
    while (j > 0 && tmp < array[j-1]) {
      array[j] = array[j-1];
      j--;
    }
    array[j] = tmp;
  }
}
```

1) Construct the control ow graph (CFG) for the method
2) Calculate the Cyclomatic complexity

Part II Function Point:

Five standard "functions"

In counting FPs there are five standard "functions" that you count. The first two of these are called Data Functions, and last three are called Transaction Functions. The names of these functions are listed below.

1. Data Functions:
    1. Internal logical files
    2. External interface files
2. Transactional Functions:
    1. External Inputs
    2. External Outputs
    3. External Inquiries

Using this terminology, when a person that counts FPs looks at a software system, they see something like this:

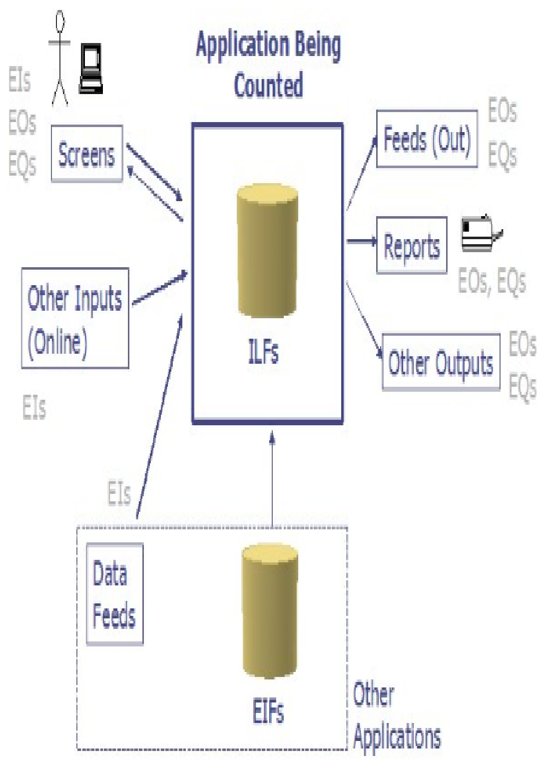


Figure 1: The view of a software application from the eyes of a Function Point practitioner.

These five functions will be discussed in greater depth in the sections that follow.

<u>The simplest way to calculate a function point count is calculated as follows:</u>

(No. of external inputs x 4) +

(No. of external outputs x 5) +

(No. of logical internal files x 10) +

(No. of external interface files x 7) +

(No. of external enquiries x 4)

Scheduling,

Activity of the Node, critical path

Project Networks

A: Activity identification (node)

ES: Earliest starting time

EC: Earliest completion time

LS: Latest starting time

LC: Latest completion time

t: Activity duration

P(A): set of predecessor nodes to node A

S(A): set of successor nodes to node A

Project Networks

- In tabular form

Activity	Predecessor	Duration
A	n/a	2
B	n/a	6
C	n/a	4
D	A	3
E	C	5
F	A	4
G	B,D,E	2

Sample Computations

ES(A) =Max{EC(j), j in P(A)}=EC(start)=0

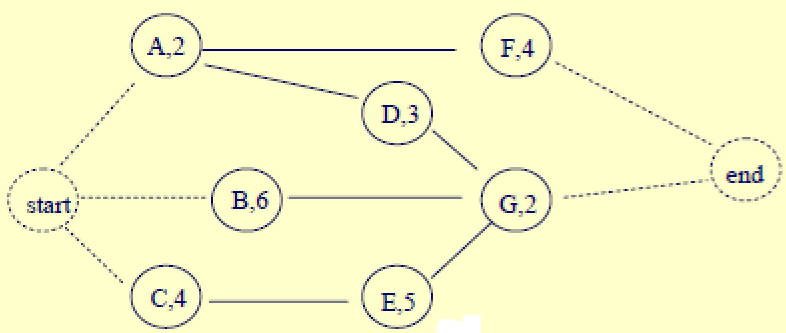
EC(A)=ES(A)+t_A=0+2=2

ES(B)= EC(start)=0

EC(B)=ES(B)+ t_B=0+6=6

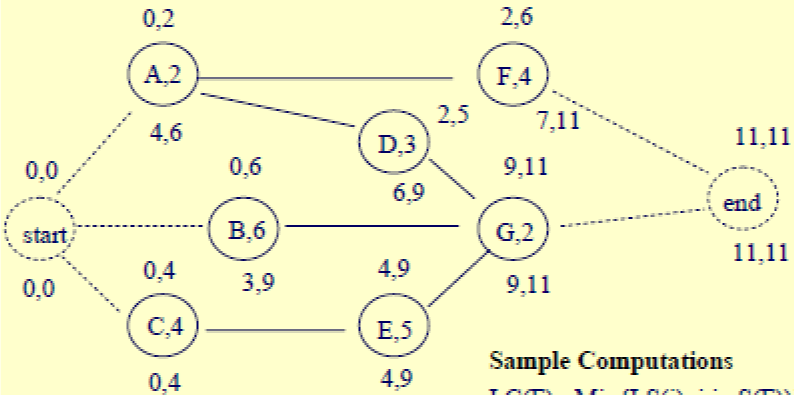
ES(F)= EC(A)=2

EC(F)= ES(F)+4=6, etc.



Project Networks

- **Notation:** Above node ES(i), EC(i), below node LS(i),LC(i)
- Zero project slack convention in force



Sample Computations

LC(F) =Min{LS(i), i in S(F))}=11

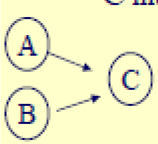
LS(F)=LC(F)-t_F=11-4=7

etc.

Project Networks

- During the **forward pass**, it is assumed that each activity will begin at its earliest starting time
- An activity can begin as soon as the last of its predecessors has finished

C must wait for both A and B to finish before it can start

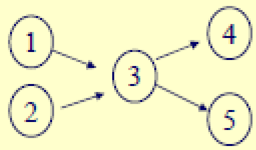


Completion of the forward pass determines the earliest completion time of the project

- During the **backward pass**, it is assumed that each activity begins at its latest completion time
- Each activity ends at the latest starting time of the first activity in the project network

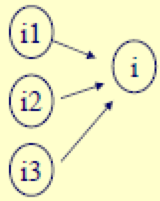
Project Networks

- **Note:**
  1=first node (activity),n=last node,i,j=arbitrary nodes, P(i)= immediate predecessors of node i, S(j)= immediate successors of node j, T_p=project deadline time



- P(3)= {1,2}
- S(3)= {4,5}

Rule 1: ES(1)=0 (unless otherwise stated)
Rule 2: ES(i)=Max j in P(i) {EC(j)}

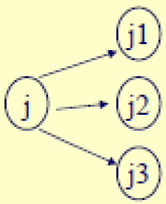


- **Why do we use "max" of the predecessor EC's in rule 2?**

Project Networks

Rule 3: EC(i)=ES(i)+t_i
Rule 4: EC(Project)=EC(n)
Rule 5: LC(Project)=EC(Project) "zero project slack convention" (unless otherwise stated for example, see Rule 6)
Rule 6: LC(Project)=Tp
Rule 7: LC(j) =Min i in S(j) LS(i)
Rule 8: LS(j)=LC(j)-tj

- **Why do we use "min" in the successor LS's in rule 7?**



Project Networks

- **Total Slack:** Amount of time an activity may be delayed from its earliest starting time without delaying the latest completion time of the project
  $TS(j)=LC(j)-EC(j)$ or $TS(j)=LS(j)-ES(j)$
- Those activities with the minimum total slack are called the critical activities (e.g., "kitchen cabinets")
- Examples of activities that might have slack
- **Free Slack:** Amount of time an activity may be delayed from its earliest starting time without delaying the starting time of any of its immediate successors.
  $FS(j)= \text{Min}_{i \text{ in } S(j)} \{ES(i)-EC(j)$
- Let's consider the sample network relative to critical activities and slack times

CPM-Determining the Critical Path AON

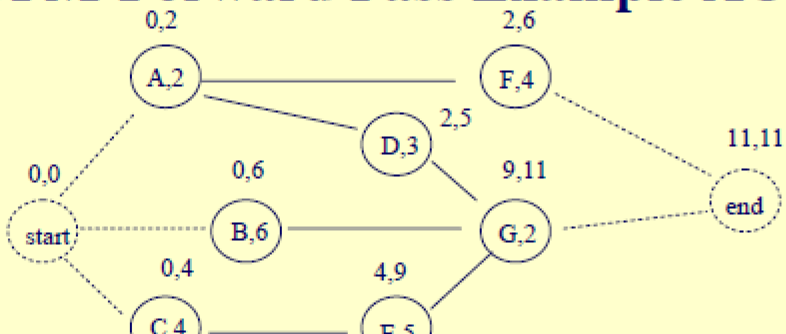
Step 1: Complete the forward pass

Step 2: Identify the last node in the network as a critical activity

Step 3: If activity i in P(j) and activity j is critical, check if EC(i)=ES(j). If yes → activity i is critical. When all i in P(j) done, mark j as completed

Step 4: Continue backtracking from each unmarked node until the start node is reached

CPM-Forward Pass Example AON



Notation: Above node ES(i), EC(i)

Activity	Predecessor	Duration
A	-	2
B	-	6
C	-	4
D	A	3
E	C	5
F	A	4
G	B,D,E	2

Sample Computations

ES(A) =Max{EC(j), j in P(A)}=EC(start)=0

EC(A)=ES(A)+t_A=0+2=2

ES(B)= EC(start)=0

EC(B)=ES(B)+ t_B=0+6=6

ES(F)= EC(A)=2

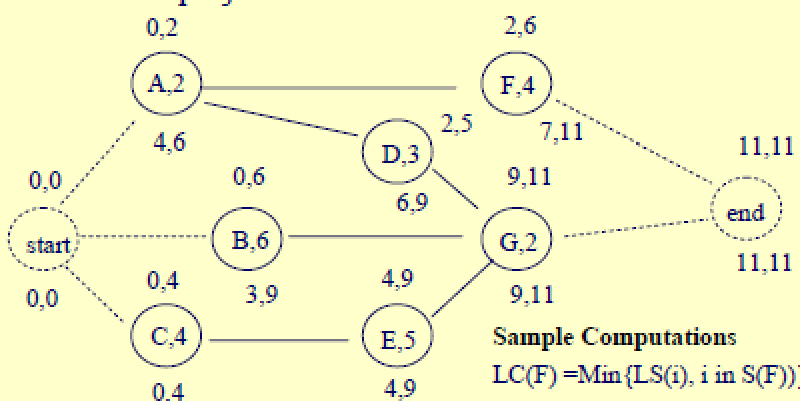
EC(F)= ES(F)+4=6, etc.

15

CPM-Backward Pass Example AON

Notation: Above node ES(i), EC(i), below node LS(i),LC(i)

- Zero project slack convention in force



Sample Computations

LC(F) =Min{LS(i), i in S(F))}=11

LS(F)=LC(F)-t_F=11-4=7

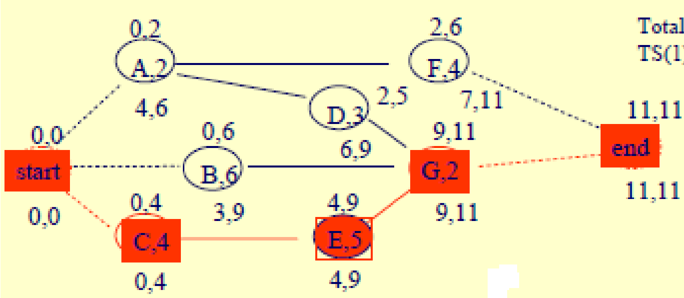
etc.

CPM-Slacks and the Critical Path AON

- **Total Slack:** Amount of time an activity may be delayed from its earliest starting time without delaying the latest completion time of the project
  TS(j)=LC(j)-EC(j) or TS(j)=LS(j)-ES(j)
- Those activities with the minimum total slack are called the critical activities.
- Examples of activities that might have slack
- **Free Slack:** Amount of time an activity may be delayed from its earliest starting time without delaying the starting time of any of its immediate successors.
  $FS(j)= \text{Min}_{i \text{ in } S(j)} \{ES(i)-EC(j)\}$
- Other notions of slack time, see Badiru-Pulat
- Let's consider the sample network relative to critical activities and slack times

CPM Analysis for Sample Network AON

Activity	Duration (Days)	ES	EC	LS	LC	TS	FS	Critical Activity?
A	2	0	2	4	6	6-2=4	Min{2,2}-2=0	No
B	6	0	6	3	9	9-6=3	Min{9}-6=3	No
C	4	0	4	0	4	4-4=0	Min{4}-4=0	YES
D	3	2	5	6	9	9-5=4	Min{9}-5=4	No
E	5	4	9	4	9	9-9=0	Min{9}-9=0	YES
F	4	2	6	7	11	11-6=5	Min{11}-6=5	No
G	2	9	11	9	11	11-11=0	Min{11}-11=0	YES



Total Project Slack
TS(1)+TS(C)+TS(E)+TS(G)+TS(n)=0