Q4.2 depth first algorithm:

1- The critical paths and completion time for each path:

a – b – f – j – k **2+6+4+4+3+4+3+6+2 = 34**

a – b – j – k **2+6+4+6+3+6+2 =29**

a – c – g – k **2+2+4+2+2+2+2=16**

a – d – h – k **2+3+5+4+5+3+2=24**

a – e – h – k **2+4+4+2+5+3+2=22**

a – e – I **2+4+4+3+4=17**

2- By ordering the paths descending :
The first path to begin with

a – b – f – j – k **2+6+4+4+3+4+3+6+2 = 34**

By executing this path k need dependencies

3- Remove the executed tasks then calculate new critical paths and the completion time for each path:

c – g – k **4+2+2+2+2=12**

d – h – k **5+4+5+3+2=19**

e – h – k **4+2+5+3+2=16**

e – I **4+3+4=11**

4- By ordering the new paths descending and start from

d – h – k 5+4+5+3+2=19

Execute d and h need dependencies.

5- Re calculate the paths

c – g – k 4+2+2+2+2=12

h – k 5+3+2=10

e – h – k 4+2+5+3+2=16

e – I 4+3+4=11

Execute: e – h

k need dependencies

6- Re calculate the paths

c – g – k 4+2+2+2+2=12

e – I 4+3+4=11

Execute: c – g

Then e – I

Then the last task K.

1- BFS:

Speed up:

Speed up = sequential / parallel

Speed up = (2+4+4+5+4+3+2+5+4+3+2)/(22)

Speed up = 38 / 22

Speed up = 1.727272727

Efficiency:

Efficiency = speed up / no.of.processor

Efficiency = 1.727272727 / 3 %

Efficiency = 57 %

2- DFS:

Speed up:

Speed up = sequential / parallel

Speed up = (2+4+4+5+4+3+2+5+4+3+2)/(23)

Speed up = 38 / 23

Speed up = 1.652173913

Efficiency:

Efficiency = speed up / no.of.processor

Efficiency = 1.652173913/ 3 %

Efficiency = 55 %