CHAPTER 3 – PLANNING AND SCHEDULING

Assumptions and Rules

- The machine scheduling problem (MSP) has an associated order-requirement weighted digraph.
- *In this class*, the times given for tasks are assumed to be given in minutes, unless you are told otherwise in the problem.
- The tasks are arranged in a ***priority list*** that is independent of the digraph.
- No processor stays idle if there is a task to be done.
- If a processor starts working on a task, the work will continue until that task is complete. No multitasking is allowed.

A task is considered ***ready*** if all its predecessors in the digraph have been completed.

List Processing Algorithm

1. **Assignment of Processors:** The lowest numbered idle processor is assigned to the highest priority ready task until either all of the processors are assigned or all of the ready tasks are being worked on.
2. **Status Check:** When a processor completes a task, that processor becomes idle. Check for ready tasks and tasks not yet completed and determine which of the following applies:
   a. If there are ready tasks, repeat Step 1.
   b. If there are no ready tasks but not every task has been completed, the idle processor remains idle until more tasks are completed.
   c. If all tasks are completed, the job is done.

Independent Tasks

When the tasks are independent, they can be performed in any order. Thus, the associated order-requirement digraph has no edges. There are different algorithms that can be used to schedule independent tasks, but we will use the List Processing Algorithm in this class.

What is the minimum time required to perform nine independent tasks with a total task time of 45 minutes on three machines if no task takes longer than 10 minutes?

A ***decreasing time priority list*** is created by listing all the tasks from the longest to the shortest completion time. If there is a tie, the lower numbered task has the higher priority.

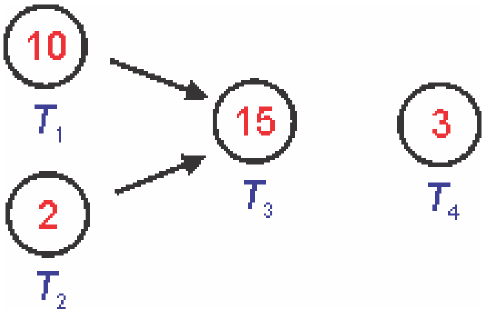
You have independent tasks of length 5, 2, 8, 1, 7, 3, 10. Using two processors, find the completion time using the list processing algorithm with

a) the priority list using the tasks in the original order and then
b) using a decreasing time priority list.

Are either of the completion times optimal?

Pizza Dinner

To make a pizza, you need to pre-heat the oven, unwrap the frozen pizza, and bake it for 15 minutes. You also need to find plates. The priority list for these tasks is T_3, T_2, T_4, T_1.



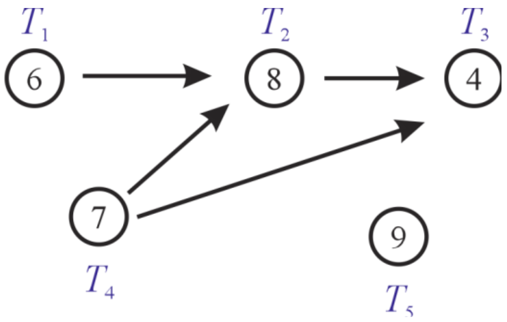
Schedule these tasks with one person working.

Will dinner get ready faster with two people? Priority List: T_3, T_2, T_4, T_1

Example:

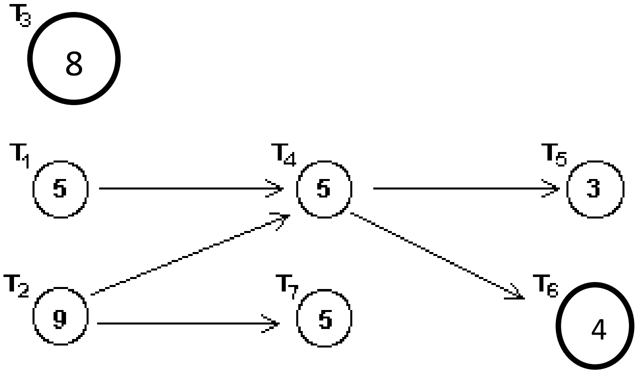
Using the order-requirement digraph below, schedule the tasks on two processors using the priority list T_5, T_4, T_3, T_2, T_1. What is the completion time if the times for each task are given in minutes? Is it optimal?

Priority List: T_5, T_4, T_3, T_2, T_1



Using the order-requirement digraph below, schedule the tasks on three processors using the priority list $T_1, T_2, T_3, T_4, T_5, T_6, T_{7.}$ What is the optimal completion time if the task times are in hours?

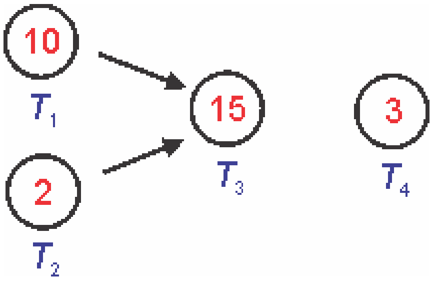
Priority List: $T_1, T_2, T_3, T_4, T_5, T_6, T_7$

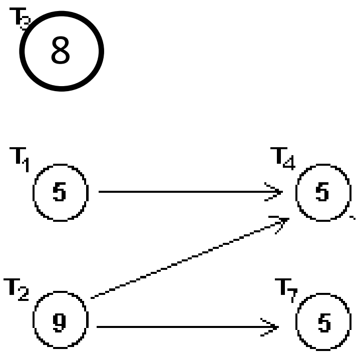


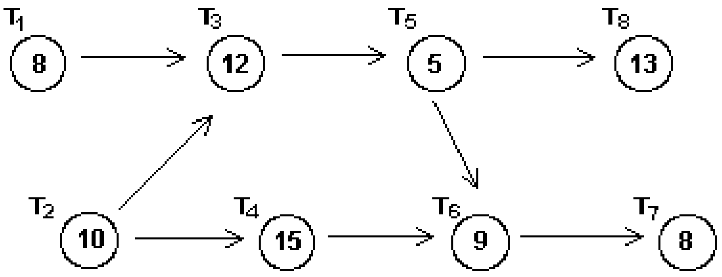
Creating a Priority List for Critical Path Scheduling

1. Find a task that heads a critical (longest) path in the order-requirement digraph. If there is a tie, chose the lowest task number.
2. Place the task found in Step 1 next in the priority list.
3. Remove the task found in Step 1 from the digraph. Remove all edges attached to the removed task to form a new diagraph.
4. If all tasks have been removed, the list is completed. If tasks remain, return to step 1.

Create a critical path priority list for the digraphs below







You have the following weights (in ounces) to pack into bins that hold no more than 9 ounces. How should this be done? What is the minimum number of bins needed?

4, 6, 1, 2, 4, 5, 1, 3, 6, 2

Next-fit Algorithm (NF): Put items into the open bin until the next item will not fit. Close the bin and open a new bin for the next item.

4, 6, 1, 2, 4, 5, 1, 3, 6, 2

First-fit Algorithm (FF): Put items into the first already open bin that has space for it. If no open bin has space, open a new bin.

4, 6, 1, 2, 4, 5, 1, 3, 6, 2

Worst-fit Algorithm (WF): Put items into an already open bin that has the most space for it. If no open bin has space, open a new bin.

4, 6, 1, 2, 4, 5, 1, 3, 6, 2

Best-fit Algorithm (BF): Put items into an already open bin that has the least space for it. If no open bin has space, open a new bin.

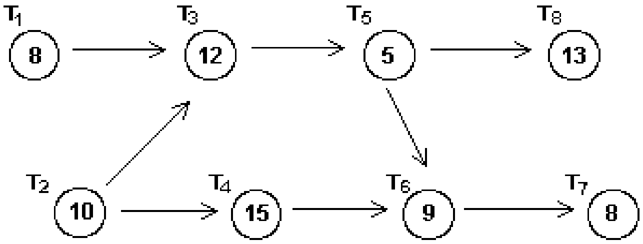
4, 6, 1, 2, 4, 5, 1, 3, 6, 2

SAMPLE EXAM QUESTIONS FROM CHAPTER 3

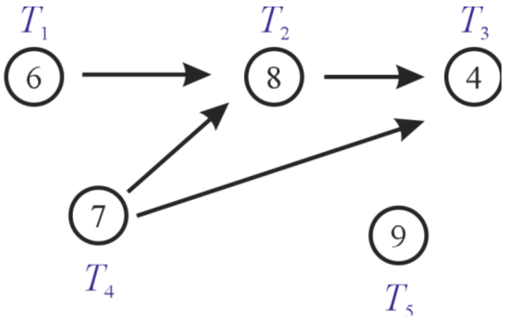
1. Given the order-requirement digraph below (with time given in minutes) and the priority list

T_4, T_5, T_6, T_7, T_8, T_1, T_2, T_3,

apply the list-processing algorithm to construct a schedule using two processors. How much time does the resulting schedule require? Is the schedule optimal? Priority List: T_4, T_5, T_6, T_7, T_8, T_1, T_2, T_3



2. Using the given graph, determine the critical path priority list.



(A) T_5, T_4, T_1, T_2, T_3

(B) T_5, T_4, T_2, T_1, T_3

(C) T_4, T_1, T_2, T_3, T_5

(D) T_4, T_1, T_2, T_5, T_3

(E) None of these/need more information

3. What is the minimum time required to perform six independent tasks with a total task time of 48 minutes on three machines if no task takes more than 12 minutes?

(A) 2 minutes
(B) 8 minutes
(C) 12 minutes
(D) 16 minutes
(E) None of these/need more information

4. Use the decreasing-time-list processing algorithm to schedule these independent tasks on two machines:

4 minutes, 5 minutes, 8 minutes, 3 minutes, 3 minutes, 7 minutes

How much time does the resulting schedule require?

(A) 15 minutes
(B) 16 minutes
(C) 17 minutes
(D) 18 minutes
(E) None of these/need more information

5. Find the packing that results from the use of given bin-packing algorithm to pack the following weights into bins that can hold no more than 8 lbs.

5 lbs, 7 lbs, 1 lb, 2 lbs, 4 lbs, 5 lbs, 1 lb, 1 lb, 3 lbs, 6 lbs, 2 lbs.

Are any of these packings optimal?

(a) Next fit

5, 7, 1, 2, 4, 5, 1, 1, 3, 6, 2

(b) First fit

5, 7, 1, 2, 4, 5, 1, 1, 3, 6, 2

(c) Worst fit

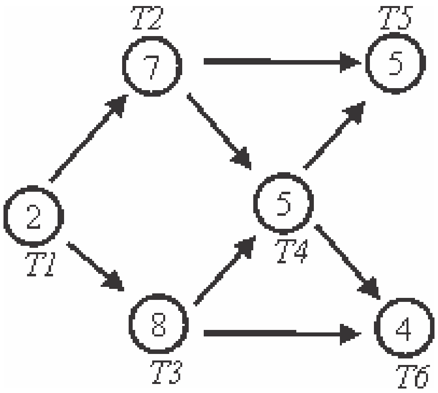
5, 7, 1, 2, 4, 5, 1, 1, 3, 6, 2

d) Best fit

5, 7, 1, 2, 4, 5, 1, 1, 3, 6, 2

6. Use the digraph below and the priority list T_6 ,T_1, T_2, T_3, T_4, T_5 to schedule these tasks on two processors. Is the schedule optimal?

Priority List: T_6 ,T_1, T_2, T_3, T_4, T_5


T2
7
T5
5
2
T1
5
T4
8
T3
4
T6
