1. Scheduling Tasks (3.1-3.3)

Definition 1.1.
A ________________is a person or machine or robot that works on a task. A **machine scheduling problem (MSP)** is to schedule tasks such that the cost needed for completing task is minimized.

In order to solve MSP, you need to take into account things such as the importance of a task and if one task needs to be completed before another task.

(1) Assumptions and Rules
- The MSP has an associated order-requirement weighted digraph.
- 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.
- In this class, the times given for tasks are assumed to be given in minutes, unless you are told otherwise in the problem.

(2) Possible Goals
- Minimize the completion time for a given number of processors.
- Minimize processor idle time.
- Minimize the number of processors needed to complete the job in a specified time.

Definition 1.2.
A task is considered ________________if all its predecessors in the digraph have been completed

Algorithm 1.3 (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.

Definition 1.4.
Independent Tasks are tasks which can be done regardless of the status of the other 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.

Example 1.5.

The task times of six independent tasks, from T_1 to T_6, are 2, 1, 5, 4, 2, 3, respectively.

a) Schedule these tasks on one processor according to the list processing algorithm with priority list T_6, T_4, T_1, T_2, T_3, T_5. What is the completion time?

b) Schedule these tasks on two processors according to the same priority list T_6, T_4, T_1, T_2, T_3, T_5. What is the completion time?

Definition 1.6.
A __________________priority list is created by listing all the tasks from the longest to the shortest completion time. In the case of a tie, the lowest numbered task is done first. This method yields relatively good schedules with independent events.

Example 1.7.
Schedule the six independent tasks, T_1 to T_6, with times 2, 1, 5, 4, 2, and 3, respectively, according to the list processing algorithm using a decreasing time priority list on two processors.

How does the completion time compare to the schedule created with two processors and the priority list T_6, T_4, T_1, T_2, T_3, T_5?

Definition 1.8.
An ________________schedule is a schedule assigning processors to tasks in such a way that it results in the shortest possible finishing time for that project with that number of processors.

- For a job consisting of all independent tasks, the optimal scheduling time can be found by comparing the time of the longest individual task and the time obtained by summing the total time of all of the tasks and dividing by the number of processors being used. The larger of these two numbers will be the optimal scheduling time.
- If you have a job where the tasks are not all independent, the optimal scheduling time will be the time it takes to complete the critical path.

Example 1.9.
What is the optimal scheduling time of the tasks above on two processors?

Example 1.10.
What is the minimum time to complete 12 independent tasks on four processors when the sum of all the times of the 12 tasks is 60 minutes and each task requires less than 10 minutes?

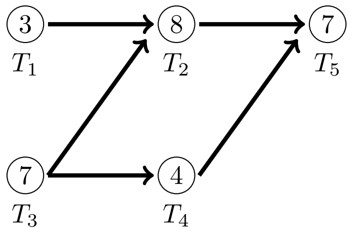
Example 1.11.
Schedule 16 different independent skits into three sessions. The times (in minutes) for the skits are given in a decreasing time priority list below.

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}
10	9	9	9	9	9	8	7	5	4	3	3	2	1	1	1

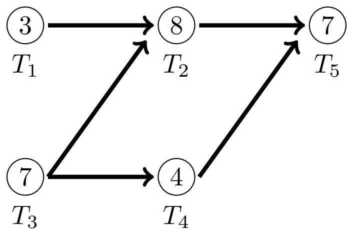
Is this optimal? Why or why not?

Example 1.12 (Nonindependent task)**.**

(a) What is the completion time for the job shown in the digraph below using one processor and the priority list T_3, T_2, T_1, T_4, T_5 ?



(b) What is the completion time for the job shown in the digraph below using two processors and the priority list T_3, T_2, T_1, T_4, T_5?



(c) Are the schedules in (a) or (b) optimal? Why or why not?

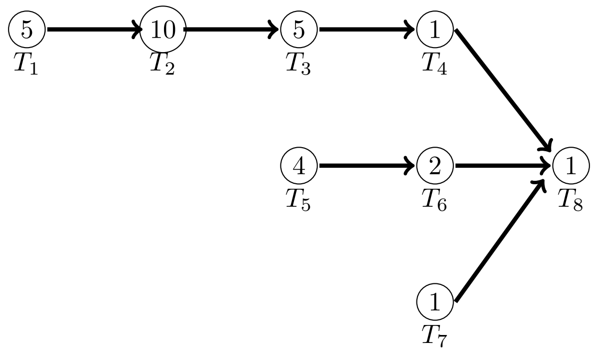
Example 1.13 (Making Fajitas)**.**

Ingredients: Tortillas, Chicken, 1 onion, Seasoning, Tomatoes, and Cilantro.

(1) Slice the onion and chicken into strips. It will take 2 minutes and 5 minutes respectively.
(2) Cook chicken 10 minutes then add onions and cook for 5 more minutes.
(3) Add seasoning (1 minute)
(4) Chop tomatoes (2 minutes) and mix with Cilantro. (1 minute)
(5) Warm tortillas (1 minute)
(6) Serve it (5 minutes)

T_1	slice chicken and onions – 5 min
T_2	cook chicken – 10 min
T_3	cook onions w/chicken – 5 min
T_4	add seasoning – 1 min
T_5	chop tomatoes – 4 min
T_6	mix tomato and cilantro – 2 min
T_7	warm tortillas – 1 min
T_8	serve – 1 min

(A) Table for tasks and time.



(B) Dependency of task as a directed graph

Apply the list processing algorithm using one and two processors to make fajitas. What are the completion times of these schedules?

One processor:
Priority list: T_6, T_5, T_1, T_2, T_3, T_4, T_8, T_7

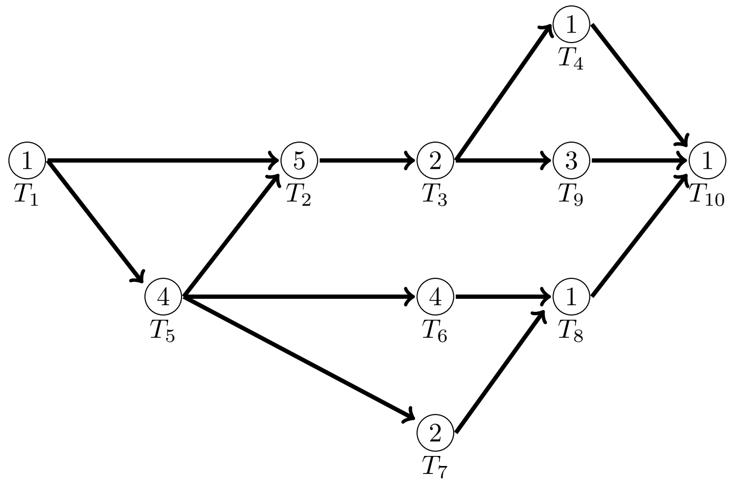
Two processor:
Priority list: T_6, T_5, T_1, T_2, T_3, T_4, T_8, T_7

Would three processors make the meal preparation any faster?

Example 1.14.

Apply the list processing algorithm to the digraph below with the priority list

$$T_1, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}, T_2, T_3$$



How long would this take with one processor?

How long would this take with two processors?

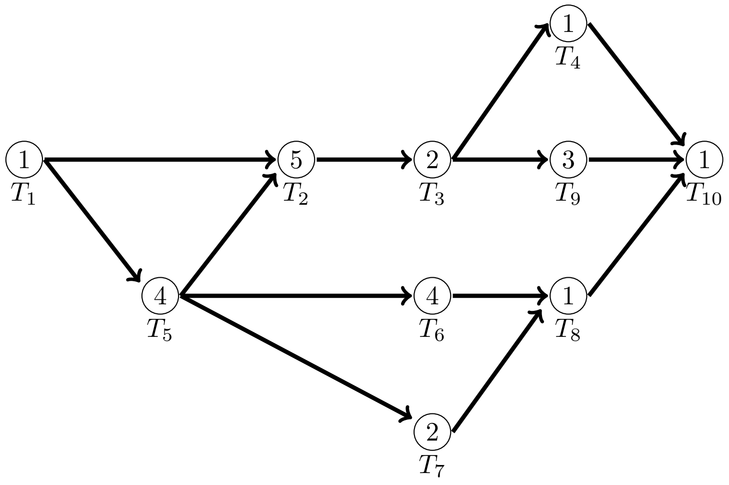
$$T_1, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}, T_2, T_3$$

What is the optimal time for completion of the job depicted in this graph?

Example 1.15.

Apply the list processing algorithm to the same digraph using the same priority list, but use three processors. How long would this take with three processors?

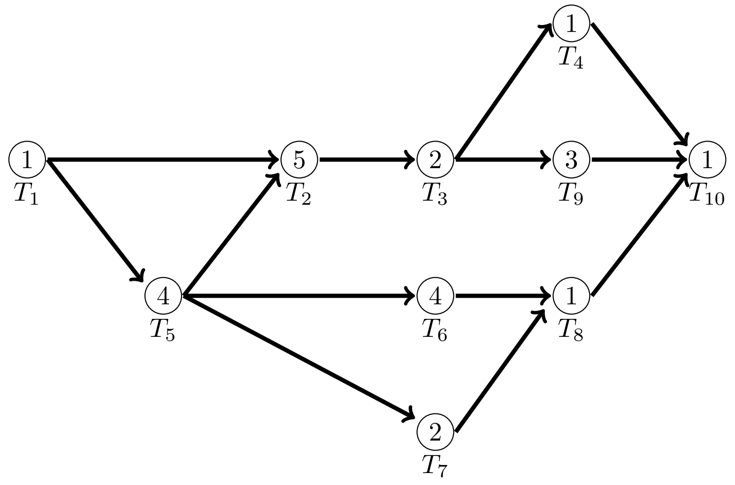
$$T_1, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}, T_2, T_3$$



Example 1.16.

Apply the list processing algorithm to the same digraph (shown below) with three processors, but with a new priority list of

$$T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}.$$



Is this schedule optimal? Why or why not?

Did changing the priority list change the solution?

Example 1.17 (Making Lunches)**.**

T_1	find lunch boxes – 5 minutes
T_2	clean lunch boxes – 3 minutes
T_3	make sandwiches – 7 minutes
T_4	wrap sandwiches – 2 minutes
T_5	wash apples – 2 minutes
T_6	put chips in bag – 3 minutes
T_7	get drink – 1 minute
T_8	pack lunch boxes – 4 minutes

Priority list: T_7, T_6, T_3, T_2, T_1, T_4, T_5, T_8

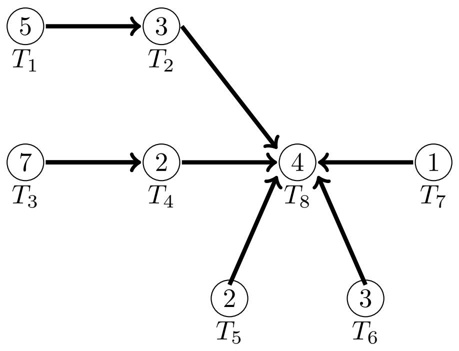
Draw the digraph showing dependency of the tasks below.

What is the optimal time to complete this process if you had an unlimited number of helpers?

Use the given priority list to schedule two processors to complete the job. Is this schedule optimal?

Use the same priority list to schedule three processors to complete the job. (T_7, T_6, T_3, T_2, T_1, T_4, T_5, T_8) Is this schedule optimal?

Example 1.18. Apply the list processing algorithm to the digraph below with the priority list T_7, T_6, T_5, T_3, T_2, T_1, T_4, T_8.



How long would this take with one processor?

What is the optimal time for completion of the job depicted in this digraph? What is the critical path?

How long would this take with three processors? T_7, T_6, T_5, T_3, T_2, T_1, T_4, T_8.

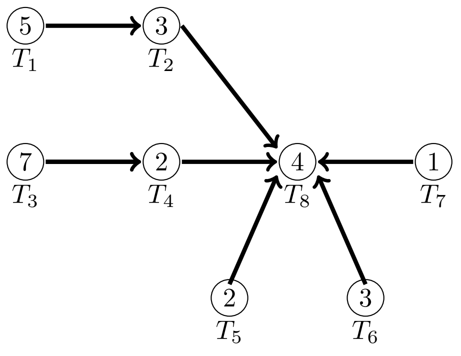
How can we create a priority list that has a reasonable chance of being optimal or near optimal?

Definition 1.19.
A **critical path scheduling** is the scheduling whose running time for completion is equal to the that of critical path.

Algorithm 1.20 (Creating a Priority List for Critical Path Scheduling)**.**

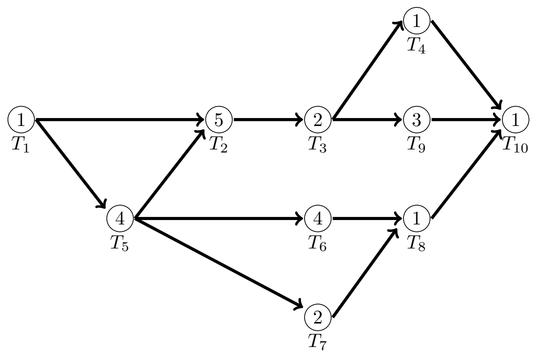
(1) Find a task that heads a critical (longest) path in the order-requirement digraph. If there is a tie, chose the lowest task number.
(2) Place the task found in Step 1 next in the priority list.
(3) Remove the task found in Step 1 from the digraph. Remove all edges attached to the removed task to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

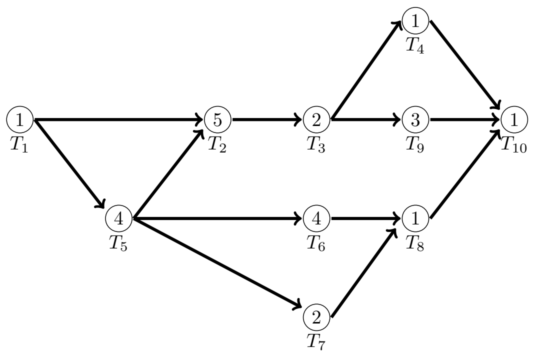
Example 1.21. Create a priority list for critical path scheduling for making lunches.

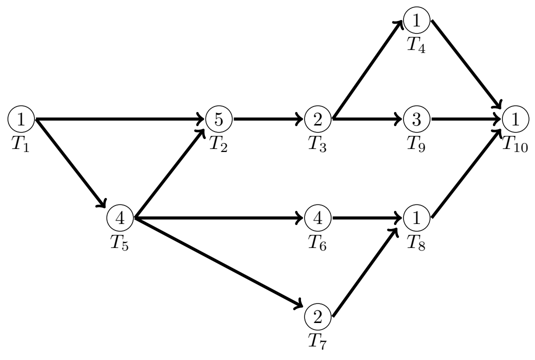


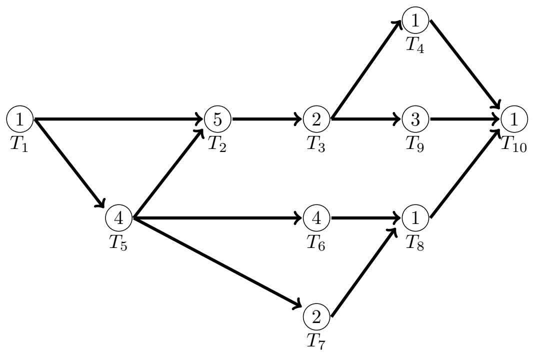
Example 1.22. Use this list to schedule two processors to complete the job. Is it optimal? If not, use three or more processors.

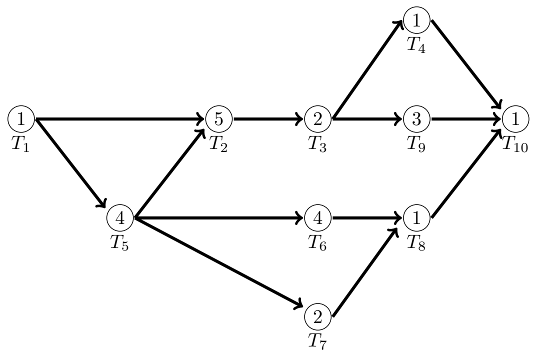
Example 1.23. Create a priority list for critical path scheduling for the digraph below.

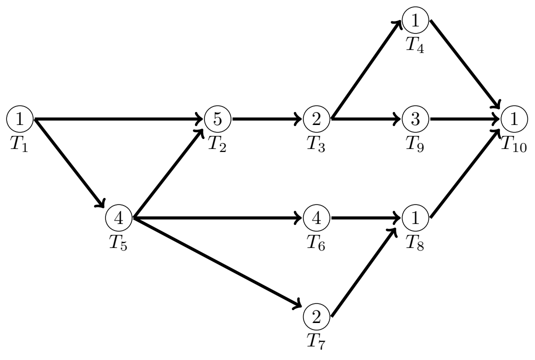


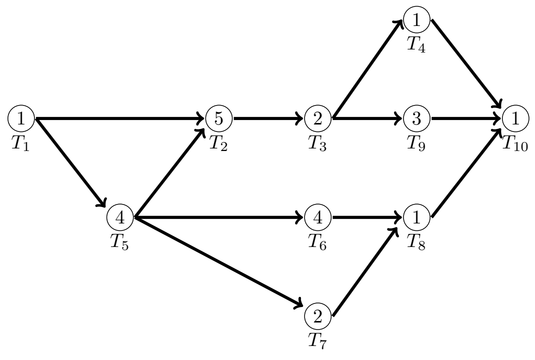


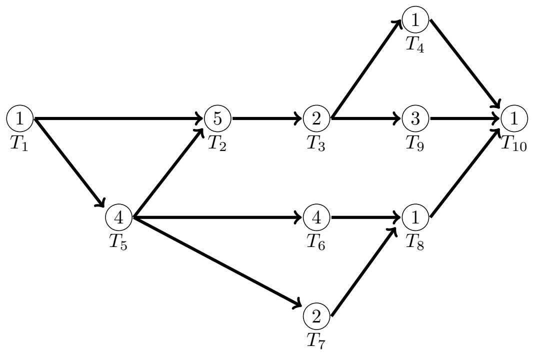












2. Bin Packing (3.4)

Example 2.1.
You have boxes available that hold at most 10 pounds and you have items that weigh 5, 7, 2, 6, 5, 1, 3, 4, 2, 3, 6, 3 pounds. What is the fewest boxes you will need?

Pack these items using the following *heuristic* methods:

Algorithm 2.2 (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.

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

Box 1	Box 2	Box 3	Box 4	Box 5	Box 6

Algorithm 2.3 (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.

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

Box 1	Box 2	Box 3	Box 4	Box 5	Box 6

Algorithm 2.4 (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.

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

Box 1	Box 2	Box 3	Box 4	Box 5	Box 6

Algorithm 2.5 (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.

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

Box 1	Box 2	Box 3	Box 4	Box 5	Box 6

Example 2.6. You have 20 books to put on shelves that have a fixed width of 12". The widths of the books, in inches, are

$$3, 7, 2, 4, 4, 4, 5, 3, 9, 5, 8, 2, 7, 8, 4, 6, 4, 1, 10, 5$$

What is the optimal number of shelves?

(a) FF: 3, 7, 2, 4, 4, 4, 5, 3, 9, 5, 8, 2, 7, 8, 4, 6, 4, 1, 10, 5

(b) NF: 3, 7, 2, 4, 4, 4, 5, 3, 9, 5, 8, 2, 7, 8, 4, 6, 4, 1, 10, 5

(c) BF: 3, 7, 2, 4, 4, 4, 5, 3, 9, 5, 8, 2, 7, 8, 4, 6, 4, 1, 10, 5

(d) WF: 3, 7, 2, 4, 4, 4, 5, 3, 9, 5, 8, 2, 7, 8, 4, 6, 4, 1, 10, 5