Math 140: Chapter 2

2.1 Scheduling - Getting Started

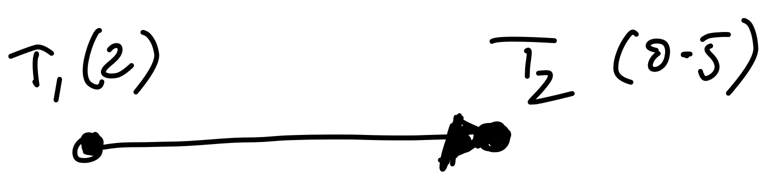
The problem of scheduling is fairly universal. Contractors need to schedule workers and subcontractors to build a house as quickly as possible. A magazine needs to schedule writers, editors, photographers, typesetters, and others so that an issue can be completed on time.

Example 1. An auto shop is converting a car from gas to electric.

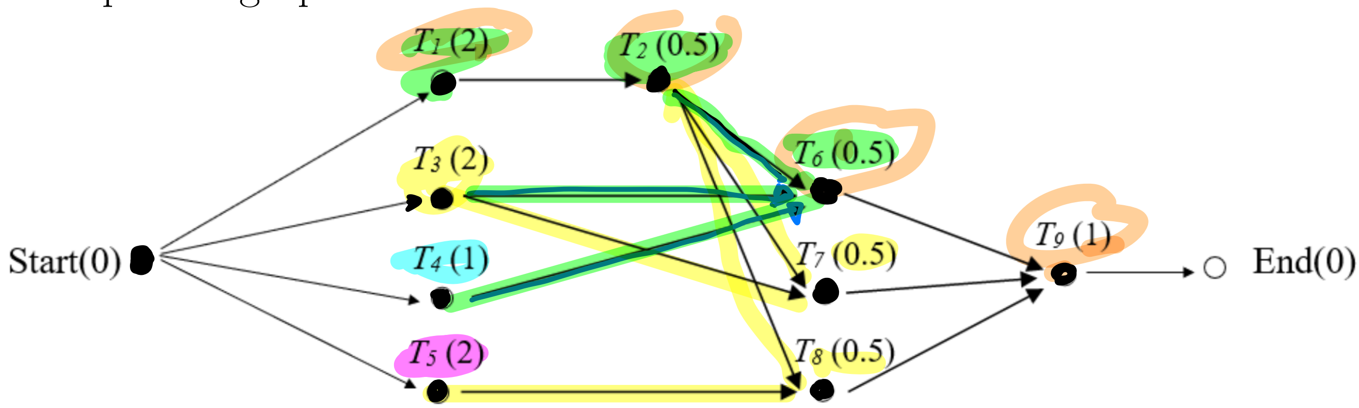
- Task 1: Remove engine and gas parts (2 days)
- Task 2: Steam clean the inside of the car (0.5 day)
- Task 3: Buy an electric motor and speed controller (2 days for travel)
- Task 4: Construct the part that connects the motor to the car's transmission (1 day)
- Task 5: Construct battery racks (2 days)
- Task 6: Install the motor (0.5 day)
- Task 7: Install the speed controller (0.5 day)
- Task 8: Install the battery racks (0.5 day)
- Task 9: Wire the electricity (1 day)

We can model this problem with a digraph.

Digraph is a directed graph in which the vertices represents tasks and the edges are arrows used to show ordering.



A complete digraph for our car conversion would look like this:



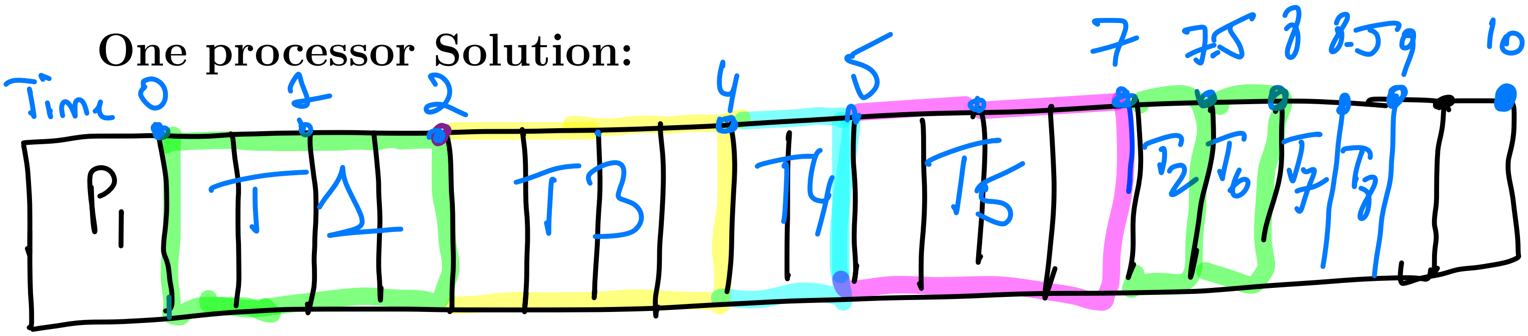
Processors are the workers completing the task

ex: Auto shop: processors are the mechanics

Assumptions:

- Every processor can do every task
- Only one processor can work at a task at a time

One processor Solution:



Finishing time w/ one processor is 10 days

Finishing time is the time that it takes to complete all tasks.
It depends on the number of processors.

For our auto example: 10 days.

Optimal schedule is the schedule with the shortest possible finishing time.

Critical Time is the absolute minimum time to complete the job, regardless of the processors working on the task.

2.2 List Processing Algorithm - LPA

To create a more procedural approach, we might begin by somehow creating a priority list.

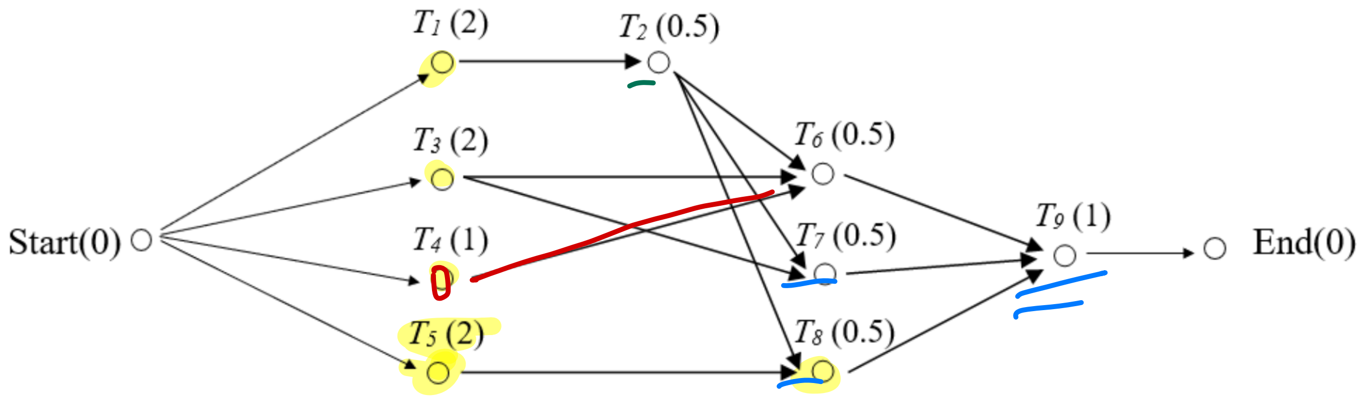
Priority List A priority list is a list of tasks given in the order in which we desire them to be completed.

Priority lists can be given to you or you have to create them.

Example 2. Schedule the project with two processors, using the priority list:

$$T_1, T_3, T_4, T_5, T_6, T_7, T_8, T_2, T_9$$

Solution for our example when we are given the priority list.



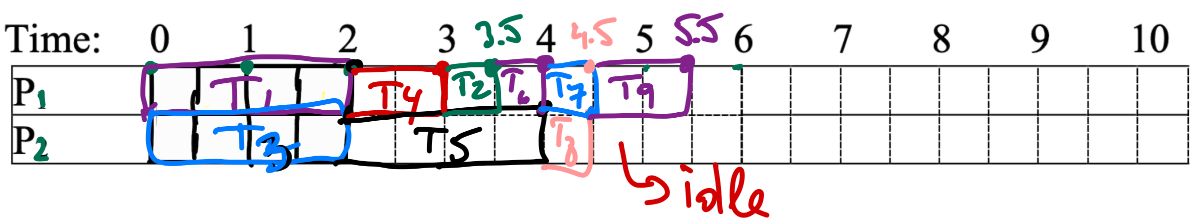
Solution for our example when we are given the priority list.

$$T_1, T_3, T_4, T_5, T_6, T_7, T_8, T_2, T_9$$

First, rewrite the priority list with the amount of time each taks takes:

T_1 (2), T_3 (2), T_4 (1), T_5 (2), T_6 (0.5), T_7 (0.5), T_8 (0.5), T_2 (0.5), T_9 (1)

Next, draw a rectangle and use as many rows as you have processors. Mark completion times above first row.



Algorithm:

Time 0: Day 0

- Mark ready tasks by circling them. in the priority list
- Assign tasks to processor and mark those tasks as in progress by slashing them.
- Mark the schedule up to here.

$$T_1, T_3, T_4, T_5, T_6, T_7, T_8, T_2, T_9$$

P_1 P_2

We jump to the next time when the next task completes, which is at time 2.

Next Time: Day 2

Update priority list.

- We mark those tasks as complete (by crossing them).
- Circle new available tasks
- Assign tasks to processor and mark those tasks as in progress by slashing them.
- We assign the next ready tasks on the list.
- Mark the schedule up to here.

Next Time: Day 3

Update priority list.

- We mark those tasks as complete (by crossing them).
- Circle new available tasks (none available)
- Assign tasks to processor and mark those tasks as in progress by slashing them.
- We assign the next ready tasks on the list.
- Mark the schedule up to here.

Next Time: Day 3.5

Update priority list.

- We mark those tasks as complete (by crossing them).
- Circle new available tasks (T_6 & T_7)
- Assign tasks to processor and mark those tasks as in progress by slashing them.

- We assign the next ready tasks on the list.
- Mark the schedule up to here.

Next Time: Day 4
Update priority list.

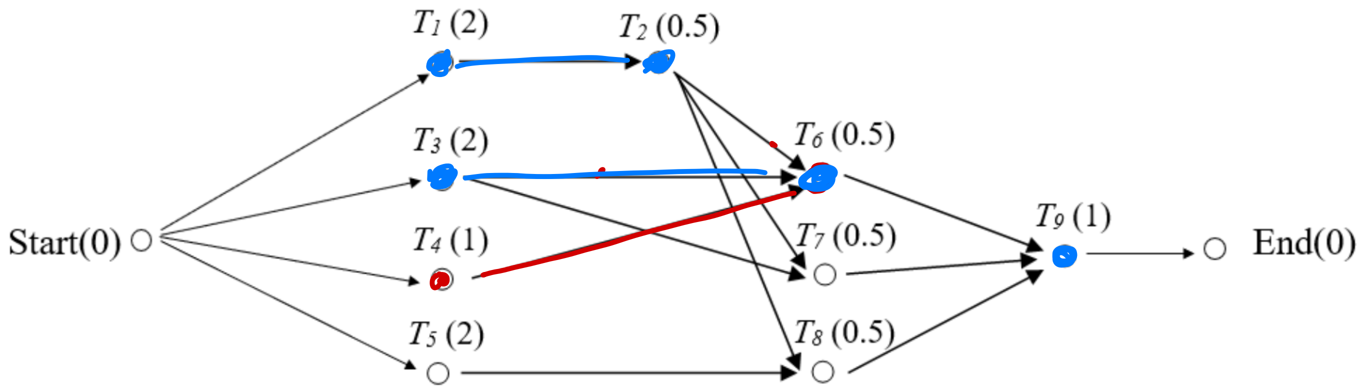
- We mark those tasks as complete (by crossing them). → T_6 & T_5
- Circle new available tasks T_8
- Assign tasks to processor and mark those tasks as in progress by slashing them.
- We assign the next ready tasks on the list.
- Mark the schedule up to here.

Next Time: ~~Day 4.5~~
Update priority list.

- We mark those tasks as complete (by crossing them).
- Circle new available tasks
- Assign tasks to processor and mark those tasks as in progress by slashing them.
- We assign the next ready tasks on the list.
- Mark the schedule up to here.

Next Time is End time for project
Day 5.5. when P_1 finishes last task
which is T_7

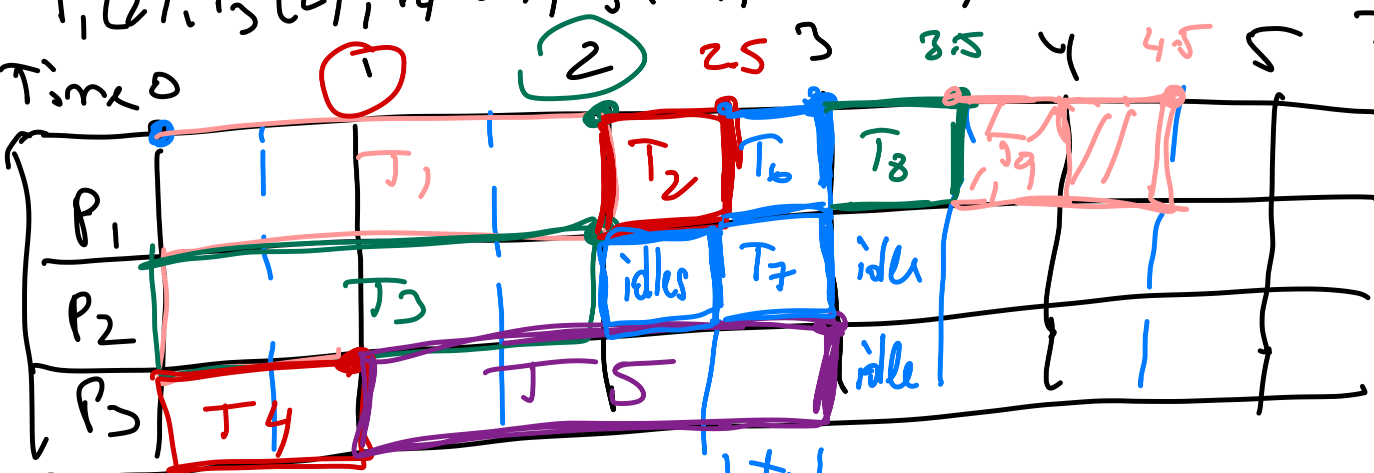
Example 3. How long does it take with 3 processors?



Use same priority list.

$$\cancel{T_1}, \cancel{T_3}, \cancel{T_4}, \cancel{T_5}, \cancel{T_6}, \cancel{T_7}, \cancel{T_8}, \cancel{T_2}, \cancel{T_9}$$

$T_1(2), T_3(2), T_4(1), T_5(2), T_6(0.5), T_7(0.5), T_8(0.5), T_2(0.5), T_9(1)$



Algorithm:

Time 0:
- none completed
- T_1, T_3, T_4, T_5 available
- $P_1 \to T_1$, $P_2 \to T_3$, $P_3 \to T_4$

Next time: Day 1
- T_4 completed
- none new available
- $P_3 \to T_5$

Next time: Day 2
- T_1 & T_3 done
- T_2
- $P_1 \to T_2$, $P_2 \to$ idles

Net time:
Day 2.5
- T_2 done
- T_6 available & T_7
- $P_1 \to T_6$ & $P_2 \to T_7$

Next time: Day 3
- T_6, T_7, T_5 done
- T_8 is open
- $P_1 \to T_8$

Next time: Day 3.5
- T_8 is done
- T_9 available
- $P_1 \to T_9$, to finish project

Finish time w/ 3 processors is 4.5 days!