# Activity or Task Scheduling Problem

This is the dispute of optimally scheduling unit-time tasks on a single processor, where each job has a deadline and a penalty that necessary be paid if the deadline is missed.

A unit-time task is a job, such as a program to be rush on a computer that needed precisely one unit of time to complete. Given a finite set S of unit-time tasks, a schedule for S is a permutation of S specifying the order in which to perform these tasks. The first task in the schedule starts at time 0 and ends at time 1; the second task begins at time 1 and finishes at time 2, and so on.

The dispute of scheduling unit-time tasks with deadlines and penalties for each processor has the following inputs:

- a set S = {1, 2, 3.....n} of n unit-time tasks.
- a set of n integer deadlines $d_1$ $d_2$ $d_3$...$d_n$ such that $d_i$ satisfies $1 \leq d_i \leq n$ and task i is supposed to finish by time $d_i$ and
- a set of n non-negative weights or penalties $w_1$ $w_2$....$w_n$ such that we incur a penalty of $w_i$ if task i is not finished by time $d_i$, and we incurred no penalty if a task finishes by its deadline.

Here we find a schedule for S that minimizes the total penalty incurred for missed deadlines.

A task is **late** in this schedule if it finished after its deadline. Otherwise, the task is early in the schedule. An arbitrary schedule can consistently be put into **early-first form**, in which the first tasks precede the late tasks, i.e., if some new task x follows some late task y, then we can switch the position of x and y without affecting x being early or y being late.

An arbitrary schedule can always be put into a **canonical form** in which first tasks precede the late tasks, and first tasks are scheduled in order of nondecreasing deadlines.

A set A of tasks is **independent** if there exists a schedule for the particular tasks such that no tasks are late. So the set of first tasks for a schedule forms an independent set of tasks 'l' denote the set of all independent set of tasks.

For any set of tasks A, A is independent if for t = 0, 1, 2.....n we have $N_t(A) \leq t$ where $N_t(A)$ denotes the number of tasks in A whose deadline is t or prior, i.e. if the tasks in A are expected in order of monotonically growing deadlines, then no task is late.

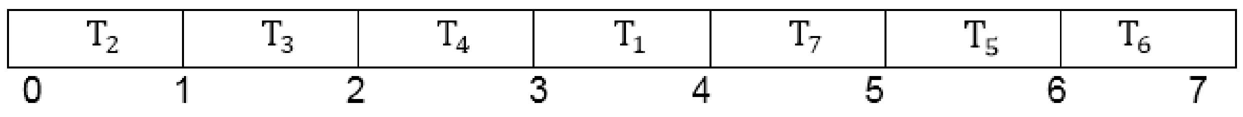
**Example:** Find the optimal schedule for the following task with given weight (penalties) and deadlines.

| | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
|---|---|---|---|---|---|---|---|
| $d_i$ | 4 | 2 | 4 | 3 | 1 | 4 | 6 |
| $w_i$ | 70 | 60 | 50 | 40 | 30 | 20 | 10 |

**Solution:** According to the Greedy algorithm we sort the jobs in decreasing order of their penalties so that minimum of penalties will be charged.

In this problem, we can see that the maximum time for which uniprocessor machine will run in 6 units because it is the maximum deadline.

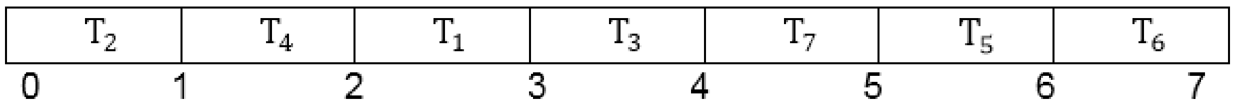
Let $T_i$ represents the tasks where i = 1 to 7



$T_5$ and $T_6$ cannot be accepted after $T_7$ so penalty is

```
w5 + w6 = 30 + 20 = 50 (2 3 4 1 7 5 6)
```

Other schedule is



(2 4 1 3 7 5 6)

There can be many other schedules but (2 4 1 3 7 5 6) is optimal.

Splunk

SPSS

Swagger tutorial

Swagger

T-SQL tutorial

Transact-SQL

Tumblr tutorial

Tumblr

React tutorial

ReactJS

Regex tutorial

Regex

Reinforcement learning tutorial

Reinforcement Learning

R Programming tutorial

R Programming

RxJS tutorial

RxJS

React Native tutorial

React Native

Python Design Patterns

Python Design Patterns

Python Pillow tutorial

Python Pillow

Python Turtle tutorial

Python Turtle

Keras tutorial

Keras

## Preparation

Aptitude

Aptitude

Logical Reasoning

Reasoning

Verbal Ability

Verbal Ability

Interview Questions

Interview Questions

Company Interview Questions

Company Questions

## Trending Technologies

Artificial Intelligence

AWS Tutorial

AWS

Selenium tutorial

Selenium

Cloud Computing

Cloud Computing

Artificial Intelligence

Hadoop tutorial
Hadoop

ReactJS Tutorial
ReactJS

Data Science Tutorial
Data Science

Angular 7 Tutorial
Angular 7

Blockchain Tutorial
Blockchain

Git Tutorial
Git

Machine Learning Tutorial
Machine Learning

DevOps Tutorial
DevOps

## B.Tech / MCA

DBMS tutorial
DBMS

Data Structures tutorial
Data Structures

DAA tutorial
DAA

Operating System
Operating System

Computer Network tutorial
Computer Network

Compiler Design tutorial
Compiler Design

Computer Organization and Architecture
Computer Organization

Discrete Mathematics Tutorial
Discrete Mathematics

Ethical Hacking
Ethical Hacking

Computer Graphics Tutorial
Computer Graphics

Software Engineering
Software Engineering

html tutorial
Web Technology

Cyber Security tutorial
Cyber Security

Automata Tutorial
Automata

C Language tutorial
C Programming

C++ tutorial
C++

Java tutorial
Java

.Net Framework tutorial
.Net

Python tutorial
Python

List of Programs
Programs