CS/ECE/ISyE 524 — Introduction to Optimization — Spring 2016

Using Optimization for Task Scheduling

Haojun Zhang (hzhang395@wisc.edu) and Keyi Cui (kcui3@wisc.edu)

Extra Models

Problem Setting

Given n tasks and m processors. For $i = 1, 2, \ldots, m$, task i has its first available time $s_i \geq 0$, deadline $t_i > 0$, duration (workload) $w_i > 0$, bonus (if completed on time) $b_i > 0$. Note that for each i, we should have $t_i - s_i \geq w_i$.

Example 1.1: Suppose s_i's and t_i's are integers, $w_i = 1$ for all i, then how can we maximize the total bonus?

This problem is taken from CS 577 Introduction to Algorithms, Spring 2016, and there is an efficient algorithm using **Dynamic Programming**. Here we will model it as an optimization problem.

Let $T = \max\{t_1, t_2, \ldots, t_n\}$. That is, we spend at most T time slots to complete tasks.

Decision Variables:

For $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, T$, define a binary variable x_{ij} to indicate whether task i is scheduled at time slot j.

Constraints:

1. For each task i, we will spend zero or one time slot. That is, $\sum\limits_{j=1}^{T} x_{ij} \leq 1$. In fact, we only need to ensure $\sum\limits_{j=s_i}^{t_i} x_{ij} \leq 1$ according to the objective function below.
2. For each time slot j, we can do zero or one task. That is, $\sum\limits_{i=1}^{n} x_{ij} \leq 1$.

Objective:

Maximize the total bonus from tasks that are completed before their deadlines. Task i is completed before deadline if $\sum\limits_{j=s_i}^{t_i} x_{ij} = 1$. Hence the objective function can be expressed as: $\sum\limits_{i=1}^{n} b_i(\sum\limits_{j=s_i}^{t_i} x_{ij})$.

Hence the model can be defined as following:

$$\begin{array}{lll}
\text{maximize} & \sum_{i=1}^{n} b_i (\sum_{j=s_i}^{t_i} x_{ij}) & \\
\text{subject to} & \sum_{j=s_i}^{t_i} x_{ij} \leq 1, & i = 1, 2, \ldots, n \\
& \sum_{i=1}^{n} x_{ij} \leq 1, & j = 1, 2, \ldots, T \\
& x_{ij} \in \{0, 1\}, & i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, T
\end{array}$$

The above model is IP. Can we relax it to a LP and solve it instead? The answer is YES.

The observation is that each variable x_{ij} can be viewed as an edge connecting task i and time slot j, then this problem is in fact a **maximum weighted bipartite matching problem**. Furthermore, we can assign zero bonus (weight) to those missing edges without affecting the objective, and now it becomes the **Assignment Problem**!

Example 1.2: In reality, tasks may have different workloads. Suppose s_i's, t_i's and w_i's are all integers, then how can we maximize the total bonus?

Can we modify previous model to solve this problem? It seems that we can use the same decision variables. For Constraint 1, all we need to modify is to replace right side with w_i. There is no need to modify Constraint 2.

But something seems missing when modifying the objective function: now task i is completed before deadline if $\sum_{j=s_i}^{t_i} x_{ij} = w_i$. Hence for each i, we need to add following logic constraint explicitly: if task i is completed before deadline, we earn bonus b_i.

Decision Variables:

1. For $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, T$, define a binary variable x_{ij} to indicate whether task i is scheduled at time slot j.
2. For $i = 1, 2, \ldots, n$, define a binary variable y_i to indicate whether task i is completed before its deadline.

Constraints:

1. For each task i, we will spend at most w_i slots. That is, $\sum_{j=s_i}^{t_i} x_{ij} \leq w_i$.
2. For each time slot, we can do zero or one task. That is, $\sum_{i=1}^{n} x_{ij} \leq 1$.
3. For each task i, we need to esure $y_i = 0$ if $\sum_{j=s_i}^{t_i} x_{ij} < w_i$. One approach is to express it as $\sum_{j=s_i}^{t_i} x_{ij} \geq w_i y_i$.

Objective:

Maximize the total bonus from tasks that are completed before their deadlines. The objective function can be expressed as: $\sum_{i=1}^{n} b_i y_i$.

Hence the model can be defined as following:

$$
\begin{array}{lll}
\text{maximize} & \displaystyle\sum_{i=1}^{n} b_i y_i & \\
\text{subject to} & \displaystyle\sum_{j=s_i}^{t_i} x_{ij} \leq w_i, & i = 1, 2, \ldots, n \\
& \displaystyle\sum_{i=1}^{n} x_{ij} \leq 1, & j = 1, 2, \ldots, T \\
& \displaystyle\sum_{j=s_i}^{t_i} x_{ij} \geq w_i y_i, & i = 1, 2, \ldots, n \\
& x_{ij}, y_i \in \{0, 1\}, & i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, T
\end{array}
$$

This is an ILP model.

Example 1.3: Now let's consider the scenario that $s_i = 0$, t_i's and w_i's are continuous constants. How should we model this problem? Note that we have restricted the first available time (s_i) to be zero for all tasks (to make this example easier).

Since now d_i's and w_i's are continuous, we cannot define variables using time slots. What should we do? First, let's review the previous examples: the solutions of models above not only give the maximized bonus, but also provide the scheduling of tasks. **Do we really need the details of scheduling? The answer is NO.**

Let $P = \{1, 2, \ldots, n\}$. The **key observation** is following: given a subset of tasks $Q \subseteq P$, if $\sum_{i \in Q} w_i > \max\{t_k : k \in Q\}$, then we cannot complete all tasks in Q on time. In other words, all tasks in Q can be completed on time if and only if for every $S \subseteq Q$, $\sum_{i \in S} w_i \leq \max\{t_k : k \in S\}$.

Let's start to model this problem. For each task i, let $S_i = \{k \in P : t_k \leq t_i\}$. If we define a binary variable y_i to indicate whether task i is completed before its deadline, then $y_i = 1$ **only if** $\sum_{k \in S_i} w_k y_k \leq t_i$.

One way to express this logic statment is to find a proper lower bound then represent it with an extra binary variable for each task i. Here we will use an alternative approach: the previous statement is equivalent to $y_i(\sum_{k \in S_i} w_k y_k) \leq t_i$, since when $y_i = 0$, it always holds due to $t_i \geq 0$. But now the right side contains quadratic combination of binary variables, so we need to add extra binary variables for each quadratic combination, similar to that learnt in class.

Decision Variables:

1. For $i = 1, 2, \ldots, n$, define a binary variable y_i to indicate whether task i is completed before its deadline.
2. For $i = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, n$, define a binary variable z_{ik} to indicate whether both tasks i and k are completed before their deadline.

Constraints:

1. for each task i, $\sum_{k \in S_i} w_k z_{ik} \leq t_i$
2. $z_{ik} = 1$ if and only if $y_i = 1$ and $y_k = 1$. That is, $y_i \geq z_{ik}$, $y_k \geq z_{ik}$, and $y_i + y_k \leq z_{ik} + 1$

Objective:

Maximize the total bonus from tasks that are completed before their deadlines. The objective function can be expressed as: $\sum\limits_{i=1}^{n} b_i y_i$.

Hence the model can be defined as following:

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} b_i y_i \\ \text{subject to} & \displaystyle\sum_{k \in S_i} w_k z_{ik} \leq t_i, \qquad i = 1, 2, \ldots, n \\ & y_i \geq z_{ik} \text{ and } y_k \geq z_{ik}, \qquad i = 1, 2, \ldots, n \text{ and } k = 1, 2, \ldots, n \\ & y_i + y_k \leq z_{ik} + 1, \qquad i = 1, 2, \ldots, n \text{ and } k = 1, 2, \ldots, n \\ & y_i, z_{ik} \in \{0, 1\}, \qquad i = 1, 2, \ldots, n \text{ and } k = 1, 2, \ldots, n \end{array}$$

This is an ILP model.

Once we figure out the subset of tasks Q to be completed on time, we can schedule them with the greedy approach: **sort tasks in Q in ascending order of deadline, then complete them in this order.**

In [ ]: