Scheduling

R. Chandrasekaran

March 12, 2020

0.1 Problems on one Machine with Due Dates:

Consider one processor (=machine) system.

Input: n jobs; p_j denotes the processing time for job j; d_j denotes the due date for job j; w_j denotes the weight for job j [indicates priority for jobs].

A schedule S indicates the manner in which these jobs are processed by the machine (not violating no-more-than-one-at-a-time rule). [Note: we are allowing preemptions *as long as they do not reduce the total duration for any job.*] Let $C_j(S)$ denote the time at which job j is completed in schedule S.

Tardiness $T_j(S)$ of job j in schedule S is given by

$$T_j(S) = \max[C_j(S) - d_j, 0]$$

There are several interesting questions:

1. Is there a schedule S in which $T_j(S) = 0$ for all j? [i.e. is there a schedule in which all due dates are met?]
2. Find a schedule S^* such that $T_{\max}(S^*) = \max_j T_j(S^*) = \min_S[T_{\max}(S)] = \min_S[\max_j T_j(\dot{S})]$
3. Find a schedule S^* that minimizes the number of jobs that tardy (i.e. whose $T_j(S^*) > 0$).
4. Find schedule S^* such that:
$$\sum_{j=1}^{n} w_j T_j(S^*) = \min_S[\sum_{j=1}^{n} w_j T_j(S)]$$
This problem is known as the weighted tardiness problem.
5. Special case of 4: All weights are equal (and hence assumed to be equal to 1]. This problem is known as the average tardiness problem ($\bar{T}$) on a single machine.

We want $\min_S[f(C_1(S), C_2(S), ..., C_n(S))]$ where $f : R^n \mapsto R$ is some performance measure. A performance measure is said to be a *regular measure* if the function f is an *isotonic* function [multi-dimensional nondecreasing function].

Theorem 1 (R. McNaughton) *If f is a regular measure, then there exists an optimal schedule in which there is no preemption.*

Proof. *Concatenate all pieces of a job (assuming the schedule uses preemption) to the last piece, sliding all pieces that occur in between to the left. By this process consolidate all jobs. No completion time is increased by this process under the assumption. Hence the result follows.* ■

All of the above mentioned problems involve regular measures. Thus, from now on, we may assume that the set of possible schedules that we need to consider are in one-to-one correspondence with the set of permutations of the set $\{1, 2, ..., n\}$. Some times we indicate a schedule S by its permutation π.

Earliest Due Date Schedule: In this schedule jobs are processed in nondecreasing order of dud dates (ties broken arbitrarily). This schedule is often denoted by EDD.

Theorem 2 *If there is a schedule for which the answer to question 1 is yes, the EDD is one of these. Moreover, an answer to question 2 is also provided by EDD.*

Proof for question 2 is provided by adjacent interchange argument. If in some schedule there are two adjacent jobs with i preceding j and $d_i > d_j$, the schedule obtained by interchanging these has a value of $T_{\max}$ no more than the one we started with.

For proof of question 1: Assume that in EDD some job is late; if there is any schedule in which all jobs are done on time the first tardy job in EDD will have to be processed earlier in this schedule – but this would imply one of jobs that is processed prior to this job will have to take its place and this will be late in that schedule.

Thus questions #1,2 can be answered in polynomial time. It turns out that question #3 can also be done in polynomial time but that is not relevant to our discussion.

Theorem 3 *Weighted Tardiness Problem is strongly NP-complete (in its decision version).*

Proof. 3-PARTITION$\leq_p$WTP.

Instance of 3-PARTITION: $|S| = \{x_1, x_2, ..., x_{3n}\}$; $\frac{B}{4} < x_i < \frac{B}{2}$;Q:Is there are partition into n disjoint sets each adding to B?

Corresponding instance of WTP: Number of jobs equals $4n - 1$. The values of (p_j, d_j, w_j) aer given as follows:

$$\begin{aligned} d_j &= 0; p_j = x_j; w_j = x_j \qquad \text{for } 1 \leq j \leq 3n \\ d_j &= (j - 3n)(B + 1); p_j = 1; w_j = 2 \qquad \text{for } 3n + 1 \leq j \leq 4n - 1 \\ V &= \frac{1}{2}(n-1)nB + \sum_{k=1}^{3n}\sum_{j=1}^{k} x_j x_k \end{aligned}$$

Q: Is there a schedule S with $[\sum_{j=1}^{n} w_j T_j(S)] \leq V$? ■

The problem is said to have agreeable weights if $[p_i < p_j] \Rightarrow [w_i \geq w_j]$. When all weights are equal to 1, the problem is an agreeably weighted problem.

The following results are taken from E. Lawler's 1977 paper on pseudopolynomial algorithms for agreeably weighted case.

Theorem 4 *Let the jobs have arbitrary weights. Let π be any sequence that is optimal for due dates $\{d_1, d_2, ..., d_n\}$. Let $C_j(\pi)$ be the completion time of job j in sequence π. Let $\{d'_1, d'_2, ..., d'_n\}$ be chosen so that*

$$\min[C_j(\pi), d_j] \leq d'_j \leq \max[C_j(\pi), d_j]$$

Then, any sequence that is optimal for due dates $\{d'_1, d'_2, ..., d'_n\}$ is also optimal for due dates $\{d_1, d_2, ..., d_n\}$. [The converse is not necessarily true].

Proof. Let π be an optimal sequence for due dates $\{d_1, d_2, ..., d_n\}$ and π' be an optimal schedule for due dates $\{d'_1, d'_2, ..., d'_n\}$. Let

$$\begin{aligned} T(\pi) &= \sum_{j=1}^{n} w_j \max[0, C_j(\pi) - d_j] \\ T'(\pi) &= \sum_{j=1}^{n} w_j \max[0, C_j(\pi) - d'_j] \\ T(\pi') &= \sum_{j=1}^{n} w_j \max[0, C_j(\pi') - d_j] \\ T'(\pi') &= \sum_{j=1}^{n} w_j \max[0, C_j(\pi') - d'_j] \end{aligned}$$

Let $\{A_1, A_2, ..., A_n\}$ and $\{B_1, B_2, ..., B_n\}$ be defined by the following relations:

$$\begin{aligned} A_j &= 0 \qquad \text{if } C_j(\pi) \leq d_j \\ B_j &= -w_j \max[0, \min[C_j(\pi'), d_j] - d'_j] \qquad \text{if } C_j(\pi) \leq d_j \\ A_j &= w_j(d'_j - d_j) \qquad \text{if } C_j(\pi) \geq d_j \\ B_j &= w_j \max[0, \min[C_j(\pi'), d'_j] - d_j] \qquad \text{if } C_j(\pi) \geq d_j \end{aligned}$$

Clearly, $A_j \geq B_j; j = 1, 2, ..., n$.

We can show that

$$\begin{aligned} T(\pi) &= T'(\pi) + \sum_{j=1}^{n} A_j \\ T(\pi') &= T'(\pi') + \sum_{j=1}^{n} B_j \end{aligned}$$

We do this term by term.

If $C_j(\pi) \leq d_j$, $\min[C_j(\pi), d_j] = C_j(\pi); \max[C_j(\pi), d_j] = d_j$. Hence

$$C_j(\pi) \leq d'_j \leq d_j$$

Hence $T_j(\pi) = w_j \max[0, C_j(\pi) - d_j] = 0 = w_j \max[0, C_j(\pi) - d'_j] = T'_j(\pi) + A_j$

If $C_j(\pi) \geq d_j$, $\min[C_j(\pi), d_j] = d_j; \max[C_j(\pi), d_j] = C_j(\pi)$. Hence

$$d_j \leq d'_j \leq C_j(\pi)$$

Hence $T_j(\pi) = w_j \max[0, C_j(\pi) - d_j] = w_j[C_j(\pi) - d_j] = w_j[C_j(\pi) - d'_j] + w_j[d'_j - d_j] = T'_j(\pi) + A_j$

Thus we have shown the first of the above equations. Now for the second:
$T_j(\pi') = w_j \max[0, C_j(\pi') - d_j]; T'_j(\pi') = w_j \max[0, C_j(\pi') - d'_j]$.
If $C_j(\pi) \leq d_j$:
We need to consider three cases;
(a) $C_j(\pi') \leq d'_j \leq d_j$: Here $T_j(\pi') = 0; T'_j(\pi') = 0; B_j = 0; T_j(\pi') = T_j(\pi') + B_j$
(b)$d'_j \leq d_j \leq C_j(\pi')$: In this case $T_j(\pi') = C_j(\pi') - d_j; T'_j(\pi') = C_j(\pi') - d'_j; B_j = w_j(d'_j - d_j); T_j(\pi') = T_j(\pi') + B_j$
(c)$d'_j \leq C_j(\pi') \leq d_j$: Here $T_j(\pi') = 0; T'_j(\pi') = C_j(\pi') - d'_j; B_j = -w_j[C_j(\pi') - d'_j]; T_j(\pi') = T_j(\pi') + B_j$
Similar arguments work for the case when $C_j(\pi) \geq d_j$.
Hence both of the equations:

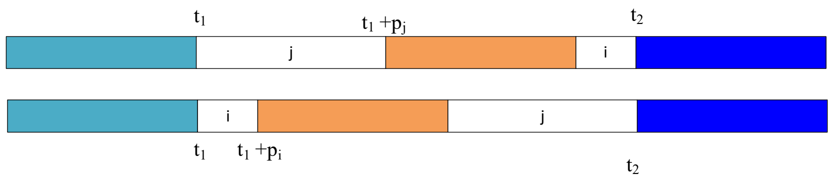
$$\begin{aligned} T(\pi) &= T'(\pi) + \sum_{j=1}^{n} A_j \\ T(\pi') &= T'(\pi') + \sum_{j=1}^{n} B_j \end{aligned}$$

hold. Using these, we can show that π' is also optimal for due dates $\{d_1, d_2, ..., d_n\}$ as follows.

Since π' is optimal for due dates $\{d'_1, d'_2, ..., d'_n\}$, it follows that $T'(\pi) \geq T'(\pi')$. Hence $T(\pi) \geq T(\pi')$ and so π' is also optimal for due dates $\{d_1, d_2, ..., d_n\}$. ■

Theorem 5 *Suppose the jobs are agreeably weighted. $\exists$ an optimal schedule π in which job i precedes job j if $d_i < d_j$ and $p_i < p_j$; moreover in π all on-time jobs are occur in nondecreasing order of due dates.*

Proof. Suppose jobs i and j satisfy above conditions and i follows j in some schedule S'. Let a schedule S be formed from S' by interchanging positions of these two jobs [Note: They may not be adjacent jobs in either schedule and we leave all other jobs in between in their old positions].



$$t_1 + p_j \leq t_2$$

Since $d_i \leq d_j; p_i < p_j$, and hence $w_i \geq w_j$it follows that

$$f(t) = w_i \max[t - d_i, 0] - w_j \max[t - d_j, 0] \geq 0 \qquad \forall t$$

and $f(t)$ is nondecreasing function of t. Hence, it follows that:

$$w_i \max[t_2 - d_i, 0] - w_j \max[t_2 - d_j, 0] \geq w_i \max[t_1 + p_j - d_i, 0] - w_j \max[t_1 + p_j - d_j, 0]$$

which implies the relation:

$$w_i \max[t_2 - d_i, 0] + w_j \max[t_1 + p_j - d_j, 0] \geq w_i \max[t_1 + p_j - d_i, 0] + w_j \max[t_2 - d_j, 0]$$

Moreover,

$$w_i \max[t_1 + p_j - d_i, 0] \geq w_i \max[t_1 + p_i - d_i, 0]$$

Combining these two relations we get:

$$w_i \max[t_2 - d_i, 0] + w_j \max[t_1 + p_j - d_j, 0] \geq w_j \max[t_2 - d_j, 0] + w_i \max[t_1 + p_i - d_i, 0]$$

This in turn implies that schedule S is better than schedule S'. The second part is easier to prove. ■

Theorem 6 *Suppose jobs are agreeably weighted and numbered in EDD order. Let* $p_k = \max_{j=1}^{n} p_j$*. Then* $\exists \delta$*,* $0 \leq \delta \leq n - k$*, such that there is an optimal schedule in which job* k *is preceded by* $[1, 2, ..., k-1, k+1, ..., k+\delta]$ *and followed by* $[k + \delta + 1, ..., n]$*.*

Proof. Let $C_k^{'}$ be the maximum completion time of job k in an optimal schedule for due dates $[d_1 \leq d_2 \leq ... \leq d_n]$. Let π be an optimal schedule with respect to due dates: $[d_1, d_2, ..., d_{k-1}, d_k' = \max[C_k^{'}, d_k], d_{k+1}, ..., d_n]$ which satisfies the conditions of the previous theorem with respect to the modified due dates. Let $C_k(\pi)$ denote the completion time of job k in π. By a previous theorem, π is also optimal for $[d_1, d_2, ...d_k, ..., d_n]$. Hence $C_k(\pi) \leq d_k^{'}$ and hence job k is on time in π for new due dates. Job k must be preceded in π by all jobs in $\{1, 2, ..., k-1, k+1, ...k+\delta\}$ where δ is the largest integer such that $d_{k+\delta} \leq d_k^{'}$. Also, job k cannot be preceded in π by any job j with $d_j > d_k^{'}$ since in that case both jobs would be one time and π is not as per previous theorem. Hence the result follows. ■

This leads to a dynamic programming algorithm which we describe now. Let jobs be agreeably weighted and numbered in EDD order. Let $p_k = \max_{j=1}^{n} p_j$. By the last theorem it follows that there is an optimal schedule in which (i) jobs $1, 2, ..., k-1, k+1, ...k+\delta$ is some order for some δ satisfying $0 \leq \delta \leq n-k$ are in front starting at time t;(ii)then we have job k with completion time

$C_k(\delta) = t + \sum_{j \le k+\delta} p_j$; (iii) followed by jobs $\{k+\delta+1, ..., n\}$ is some order starting at time $C_k(\delta)$. Let $S(i,j,k) = \{r : i \le r \le j; p_r < p_k\}$ and $T(S(i,j,k),t) =$ the the total weighted tardiness for an optimal schedule for the set of jobs $S(i,j,k)$ starting at time t. The DP recurrence relation using the last theorem is given by:

$$T(S(i,j,k),t) = \min_{\delta}[T(S(i,k+\delta,k'),t) + w_k \max[C_{k'}(\delta) - d_{k'}, 0] + T(S(k'+\delta+1,j,k'), C_{k'}(\delta))$$

where k' satisfies the relation $p_{k'} = \max\{p_j : j \in S(i,j,k)\}$ and $C_{k'}(\delta) = t + \sum_{j \in S(i,k+\delta,k')} p_j$. Basis cases are: $T(\phi,t) = 0; T(\{j\},t) = w_j \max[t + p_j - d_j, 0]$. The number of these set of the form $S(i,j,k)$ is $O(n^3)$. There are no more than $P = \sum_{j=1}^n p_j \le np_{\max}$ values of t that we need to consider. Hence the algorithm is $O(n^4 P) = O(n^5 p \max)$. Hence we have a pesudopolynomial algorithm for this problem.

0.1.1 FPTAS for $\overline{T}$:

Let $T^* = \min_\pi \sum_{j=1}^n T_j(\pi); T_{\max} = \min_\pi \max_j T_j(\pi); T_{EDD} = \sum_{j=1}^n T_j(EDD)$. It is easy to show that:

$$T_{\max} \le T^* \le T_{EDD} \le nT_{\max}$$

Let $T(S.t)$ denote the *maximum* total tardiness for a subset S of jobs starting at time t. For any given set S there is an easily computed time t^* such that

$$T(S,t) = \begin{cases} 0 & t \le t^* \\ > 0 & t > t^* \end{cases}$$

Moreover, it is easy to show that $T(S, t^* + \Delta) \ge \Delta$ for $\Delta \ge 0$. Hence, we need compute $T(S,t)$ only for $t^* \le t \le nT_{\max}$. Hence we can replace in the dynamic programming algorithm P by $nT_{\max}$ and hence get an algorithm whose complexity is $O(n^5 T_{\max})$.

Easier instance: Replace p_j by $p'_j = \left\lfloor \frac{p_j}{K} \right\rfloor K \le p_j < p'_j + K$. Let T'_π denote total tardiness for the optimal schedule π for $\{p'_j, d_j\}$ [or equivalently $\frac{1}{K}T'_A$ be total tardiness for the optimal schedule for $\{\left\lfloor \frac{p_j}{K} \right\rfloor, \frac{d_j}{K}\}$– note that the optimal schedule for both of these instances is the same since the data are multiples of each other]. It is easy to see that:

$$T'_\pi \le T^* \le T_\pi < T'_\pi + K\frac{n(n+1)}{2} \le T^* + K\frac{n(n+1)}{2}$$

Hence, we have:

$$T_\pi - T^* \le K\frac{n(n+1)}{2} \le \epsilon T_{\max} \le \epsilon T^*$$

if we set $K = \frac{2\epsilon}{n(n+1)} T_{\max}$. Hence, $T_\pi \leq (1+\epsilon)T^*$. If we solve the problem $\{\lfloor \frac{p_j}{K} \rfloor, \frac{d_j}{K}\}$, time bound is $O(n^5 \frac{T_{\max}}{K}) = O(n^7 \frac{1}{\epsilon})$. Hence we have FPTAS for this problem.

References:

1. A Pseudopolynomial Algorithm for Sequencing Jobs to Minimize Total Tardiness, by E. Lawler, Annals of Discrete Mathematics, 1 (1977), pp. 331-342

   A Fully Polynomial Approximation scheme for the Total Tardiness Problem, by E. Lawler, O.R. Letters, 1, #6, 1982, pp. 207-8.

   Minimizing Total Tardiness on One Machine is NP-hard, by J. Du and J.Y.T. Leung, Mathematics of Operations Research, 15, (1990), pp. 483-495.

3-D Matching:

Input: Given three disjoint sets W, X, Y with $|W| = |X| = |Y| = q$, a subset $S \subseteq W \times X \times Y$.

$M \subseteq S$ is called a three dimensional matching $[(w_1, x_1, y_1) \in M, (w_2, x_2, y_2) \in M] \Rightarrow [w_1 \neq w_2; x_1 \neq x_2; y_1 \neq y_2]$. Size of M is the number of triplets in M.

Question: Is there a matching M of size equal to q?

Theorem 7 *3-D Matching is NP-complete. [Pages 50-53 in Garey-Johnson Book]*

3-PARTITION:

Input: A set S with $|S| = \{x_1, x_2, ..., x_{3n}\}$ of numbers [possibly same number repeated many times].

Question: Can we partition S into $S_1, S_2, ..., S_n$ with $|S_j| = 3$ for $j = 1, 2, ..., n$ and $\sum_{i \in S_j} x_i = \frac{\sum_{i=1}^{3n} x_i}{n}$ for all $j = 1, 2, ..., n$?

The particular case that is interesting is (still NP-complete) is when $\frac{B}{4} < x_i < \frac{B}{2}$ where $\sum_{i=1}^{3n} x_i = nB$. In this case each S_j must contain exactly 3 elements to add up to B. [This is where the name comes from.]

Theorem 8 *3-PARTITION is strongly NP-complete [i.e. NP-complete even if the data are polynomially bounded when expressed in unary.] [Pages 96-103 in Garey-Johnson Book]*