(6) LP Techniques

$1|prec|\sum w_jC_j$ (and $1|prec, \{r_j, pmtn\}|\sum w_jC_j$)

$1|prec|\sum w_jC_j$ is NP-hard (and hence so is $1|prec, \{r_j, pmtn\}|\sum w_jC_j$). In fact, even the non-weighted problem $1|prec|C_j$ is NP-hard

Our goal is to write an LP-relaxation for $1|prec|\sum w_jC_j$, with C_j variables denoting the completion times of jobs, and work our way towards an approximation algorithm.

Notice that the basic set of constraints need not make a valid schedule:

i.e.

$$\min \sum_j w_jC_j \qquad (J = \text{set of all jobs, } |J| = n)$$

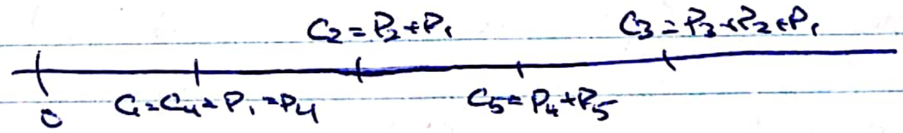
s.t.

$$C_k \geq C_j + p_k \qquad \text{if } j \to k$$

$$C_j \geq p_j \qquad \forall j \in J$$

But the following is a feasible solution:

$1 \to 2 \to 3, \quad 4 \to 5$



which is clearly not a valid schedule.

Continued

Consider:

(LP)

$$\min \sum_{j\in J} w_j C_j$$

s.t.

$$C_k \geq C_j + P_k \quad \text{if } j \to k \qquad (1)$$

$$C_j \geq P_j \quad \forall j \in J \qquad (2)$$

$$\sum_{j\in A} P_j C_j \geq \frac{1}{2} P(A)^2 + \frac{1}{2} P^2(A) \quad \forall A \subseteq J \qquad (3)$$

where:

$$P(A) = \sum_{j\in A} P_j \quad \text{and} \quad P^2(A) = \sum_{j\in A} P_j^2$$

Lemma 6.1:

Let OPT be the optimal value of $1|prec|\sum w_j C_j$ and OPT_{LP} be the optimal value of (LP). Then,

$$OPT_{LP} \leq OPT.$$

[Proof]

We want to show that every feasible schedule yields a feasible solution to (LP), where $C_j = C_j^S \ \forall j$

Clearly the C_j's will satisfy (1) and (2).

Now, consider any $A \subseteq J$.

Continued.

[Proof] (con't)

We will say $k \leq j$ if k is [redacted] schedule before j in S.

By def'n:

$$C_j^S = C_j = \sum_{k \in S: k \leq j} p_k \geq \sum_{k \in A: k \leq j} p_k$$

Then,

$$\sum_{j \in A} p_j C_j \geq \sum_{j \in A} \sum_{\substack{k \in A: \\ k \leq j}} p_j p_k$$

$$= \sum_{j \in A} p_j^2 + \sum_{\substack{j,k \in A \\ k < j}} p_j p_k$$

$$= \frac{1}{2} \sum_{j \in A} p_j^2 + \frac{1}{2} \left(\sum_{j,k \in A} p_j p_k \right)$$

$$= \frac{1}{2} p^2(A) + \frac{1}{2} p(A)^2 \qquad \square$$

Using the LP, we can consider the following algorithm:

Algorithm A_1:

(1) Solve (LP) to obtain solution $\{C_j^*\}$
 (Note: These C_j^*'s need not correspond to job completion times in a feasible schedule)

(2) Schedule jobs in increasing order of C_j^*
 (Note: Schedule returned will be feasible, by design of the (LP): If $j \to k$, then $C_j^* \leq C_k^*$)

$\hookrightarrow$ Theorem 6.1

Theorem 6.1:
A is a 2-approx. alg. for $1|prec|\sum w_j C_j$
[Proof]
Let jobs be ordered $1, \ldots, n$ so that $C_1^* \leq \ldots \leq C_n^*$ and let S be the schedule outputted by A.

We want to show: $C_j^S \leq 2C_j^* \quad \forall j=1,\ldots,n.$
So that:

$$\sum w_j C_j^S \leq 2 \sum w_j C_j^*$$
$$\Rightarrow \quad OPT_{LP} \leq 2 \cdot OPT_{1|prec|\sum w_j C_j}$$

Fix job j. Let $A = \{1, \ldots, j\}$ and consider (3) for job-set A:

(i) $\sum_{k \in A} p_k C_k^* \geq \frac{1}{2} p(A)^2 + \frac{1}{2} p^2(A) \geq \frac{1}{2} p(A)^2$

(ii) $C_j^* \cdot \sum_{k \in A} p_k \geq \sum_{k \in A} p_k C_k^*$ (since $C_k^* \leq C_j^*$)

$$\Rightarrow C_j^* \cdot \sum_{k \in A} p_k \geq \frac{1}{2} p(A)^2$$

$$\Rightarrow C_j^* \geq \frac{1}{2} p(A)^2 = \frac{1}{2} \sum_{k \leq j} p_k = \frac{1}{2} C_j^S$$

□.

Q: Is A efficient? In particular, is solving (LP) efficient?
- Notice that if $|J| = n$, then there are 2^n subsets $A \subseteq J$, and hence 2^n constraints. However, in fact:

Theorem 6.2:
(LP) can be solved in polytime
[Proof] See Theorem 6.8

$1|prec, r_j|\sum w_jC_j$ and $1|prec, r_j, pmtn|\sum w_jC_j$

Claim 6.3:
We may assume WLOG that if $j \to k$, then $r_k \geq r_j + p_j$

[Proof]
The earliest time k can start is $r_j + p_j$, so if $r_k < r_j + p_j$, we can always advance the release date of k to $r_k \leftarrow \max(r_k, r_j + p_j)$ without affecting the feasibility of the schedule □

As with $1|prec|\sum w_jC_j$, we will write an LP-relaxation for our problem.

(LP')

min $\sum w_jC_j$

s.t.

$$C_k \geq C_j + p_k \qquad \forall j \to k \qquad (1)$$

$$C_j \geq r_j + p_j \qquad \forall j \qquad (2)$$

$$\sum_{j \in A} p_jC_j \geq p(A)r(A) + \frac{1}{2}p(A)^2 + \frac{1}{2}p^2(A) \qquad \forall A \subseteq J \qquad (3)$$

where:

$r(A) := \min_{j \in A} r_j$ for all $A \subseteq J$.

Note:
It is easy to see:

$$OPT_{LP'} \leq OPT_{1|prec, r_j, pmtn|\sum w_jC_j} \leq OPT_{1|prec, r_j|\sum w_jC_j}$$

↳ Algo.

Algorithm A_2 (for $1|prec, r_j, pmtn|\sum w_j C_j$)
1) Solve (LP') to obtain $\{C_j^*\}$
2) Schedule preemptively in increasing C_j^* order

(i.e. At each point of time, schedule the available job with smallest C_j^* value, preempting as necessary)

Theorem 6.4:
A_2 produces a feasible schedule
[Proof]
We want to ensure that precedence constraints are respected.
Suppose $j \to k$. By Claim 3, if k is available at time t; either:
- j is also available at time t, or
- j is completed by time t

And also, $C_j^* \leq C_k^*$. by our ordering, unless j has completed, we will not process k. □

Theorem 6.5:
A_2 is a 2-approx. for $1|prec, r_j, pmtn|\sum w_j C_j$.
[Proof]
Let S be our schedule outputted by A_2, and order jobs so that:

$$C_1^S \leq C_2^S \leq \ldots \leq C_n^S$$

We want to show that: $C_j^S \leq 2 \cdot C_j^* \quad \forall j$

Go Con't

[Proof] (con't)

Consider job j and let t be the earliest time before C_j^S s.t.:

1) There is no idle time in $[t, C_j^S]$, and
2) m/c is only processing jobs k s.t. $k \le j$ in $[t, C_j^S]$

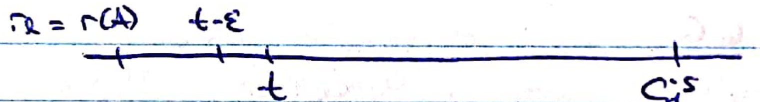
Let $A = \{\text{jobs } k : k \text{ scheduled in } [t, C_j^S]\}$

By def'n: $\forall k \in A$, $C_k^* \le C_j^*$

We claim: $\boxed{t = r(A)}$.

Clearly $t \ge r(A)$, since some job $k \in A$ is scheduled at time t.

Suppose $t > r(A)$, and suppose $l \in A$ s.t. $r_l = r(A)$.



At time $t-\varepsilon$, l is available since $r_l \le t-\varepsilon$ and l has not completed since l is scheduled in $[t, C_j^S]$. So, at time $t-\varepsilon$ the m/c cannot be idle, and either:

- has scheduled l or some $l' \le l$, but then we can move l to an earlier time
- has scheduled some $l' > j$, contradicting our schedule rule.

$\hookrightarrow$ con't

[Proof] (Cont)

So, $t = r(A)$. Applying (3) in (LP'), we get:

- $\sum_{k \in A} p_k C_k^* \geq p(A) r(A) + \frac{1}{2} p(A)^2 + \frac{1}{2} p^2(A)$

$$\geq p(A) r(A) + \frac{1}{2} p(A)^2$$

- $C_j^* p(A) \geq \sum_{k \in A} p_k C_k^*$

(Since $C_k^* \leq C_j^*$)

$$\Rightarrow \quad C_j^* p(A) \geq p(A) r(A) + \frac{1}{2} p(A)^2$$

$$\Rightarrow \quad C_j^* \geq r(A) + \frac{1}{2} p(A)$$

So,

$$C_j^S \leq t + p(A) = r(A) + p(A) \leq 2 C_j^* \quad \square$$

Proposition 6.6:

A_2 + CONVERT gives a 4-approx $1|prec, r_j|\sum w_j C_j$

Algorithm A_3:

1) Solve (LP') to obtain $\{C_j^*\}$
2) Non-preemptively schedule in increasing C_j^* order

Theorem 6.7:

A_3 is a 3-approx. algo. for $1|prec, r_j|\sum w_j C_j$.

Proof.

[Proof]
Let S be the schedule produced by A_3 and order jobs: $C_1^* \leq \dots \leq C_n^*$ (So, $C_1^S \leq \dots \leq C_n^S$)

Let $A = \{1, \dots, j\}$, then by (3) from (LP'), we get:

$$p(A) C_j^* \geq \sum_{k \in A} p_k C_k^* \geq \frac{1}{2} p(A)^2$$

$$\Rightarrow \quad p(A) \leq 2C_j^*$$

So,

$$C_j^S \leq \max_{k \leq j} r_k + \sum_{k \leq j} p_k$$

$$\leq C_j^* + 2C_j^* \quad (\text{since } r_k \leq C_k^* \leq C_j^*)$$

$$= 3C_j^* \quad \forall j$$

Which gives:

$$\sum w_j C_j^S \leq 3 \cdot \sum w_j C_j^*$$

$$= 3\, OPT_{(LP')}$$

$$\leq 3 \cdot OPT_{1|prec, r_j|\sum w_j c_j}$$

□.

Ellipsoid Method:
First polytime algorithm to solve LPs

Def'n 6.1: (Separation Oracle)
For an LP with feasible region:

$$K := \{x \in \mathbb{R}^n : a_i^T x \leq b \ \ \forall i = 1, \dots, m\}$$

Given input $y \in \mathbb{R}^n$, the separation oracle is a procedure that correctly determines if $y \in K$, or finds a constraint $a_i^T x \leq b_i$ of K that is violated by y.

Theorem 6.8 (Ellipsoid Method)

Consider an LP:

(P)

$$\min\ c^T x$$

s.t.

$$a_i^T x \le b_i \quad \forall i = 1, \dots, m$$

$$x \in \mathbb{R}^n$$

Let $S := \max(\text{size of } c, \max_{i=1,\dots,m}(\text{size of } (a_i, b_i)))$

Given a separation oracle A for (P), we can solve (P) using poly(n, S) calls to A and poly(n, S) [redacted] of other operations.

Corollary 6.8:

Given polytime separation oracle for $\{x \in \mathbb{R}^n : a_i^T x \le b_i,\ i = 1, \dots, m\}$, we can solve (P) in poly(n, S) time.