Homework 1

40.302 Advanced Topics in Optimization, Student ID 1002011

Yustynn Panicker

February 11, 2019

Contents

1 Question 1

1.1 Question 1a

Optimal Sequence: {1, 2, 3, 4}

$L_{max} = 6$

This was obtained using the general rule below.

1.2 Question 1b, 1c, 1d

A general rule for constructing an optimal schedule for any problem $1|p_j, d_j|L_{\max}$ is to sequence the jobs in increasing order of d_j. I'll give a proof for this, which covers question (b). Question (c) therefore has no valid argument. Question (d) is trivially rule (iii), assuming my argument holds.

I know that it was mentioned in class that there is no general rule for this problem, but I am unable to find any flaw with this.

1.2.1 Proof by Adjacent Pairwise Interchange Argument

1.2.1.1 Assuming the Opposite Proposition We assume that ordering the job sequence by ascending d_j does not result in an optimal strategy.

Therefore, for some schedule S, $\exists j_i, j_j$ where j_j is processed before j_i such that:

1. j_i and j_j are processed adjacent to each other
2. $d_i < d_j$
3. Swapping the order of processing the j_i, j_j to create schedule S' results in $L^{S'}_{\max} > L^{S}_{\max}$

Put simply, there exists some case where processing a job with an earlier due date first makes the objective function worse for some schedule(s).

Note that swapping the jobs clearly has no effect on the lateness of jobs that come before or after this pair. This is evident from the total time taken for this segment of processing being constant.

1.2.1.2 Set-Up

1.2.1.2.1 Segment Processing Time Let t be the sum of $p_j + p_i$ and any unforced idleness between the processing of the two jobs

1.2.1.2.2 Relation between C^S_j and $C^{S'}_i$ As the starting point for j_j under S is the same as that of j_i under S' due to the swap,

$$C_j^S - p_j = C_i^{S'} - p_i$$
$$C_i^{S'} = C_j^S + p_i - p_j$$

This result is used later, when comparing L_i^S and $L_j^{S'}$

1.2.1.2.3 Proof that $L_i^S > L_j^S$

$$L_j^S = C_j^S - d_j$$
$$L_i^S = C_j^S + t - p_j - d_i$$

$t > p_j$ (t includes p_j) and $d_i < d_j$, $\implies L_i^S > L_j^S$

1.2.1.2.4 Proof that $L_j^{S'} > L_i^{S'}$

$$L_i^{S'} = C_i^{S'} - d_j$$
$$L_j^{S'} = C_i^{S'} + t - p_i - d_i$$
$$= C_j^S + t - p_j - d_j$$

$t > p_i$ (t includes p_i) and $d_i < d_j$, $\implies L_j^{S'} > L_i^{S'}$

1.2.1.3 Proof that $L_i^S > L_j^{S'}$

$$L_i^S - L_j^{S'} = (C_j^S + t - p_j - d_i) - (C_j^S + t - p_j - d_j)$$
$$= d_j - d_i$$
$$> 0 \text{ as } d_j > d_i$$

1.2.1.4 Implication of $L_i^S > L_j^{S'}$ Part of what the opposite proposition implied was that swapping the order of processing the j_i, j_j to create schedule S' results in $L_{\max}^{S'} > L_{\max}^S$

However, this is clearly impossible as the maximum lateness among the two jobs has strictly decreased.

$\therefore L_{\max}^{S'} \leq L_{\max}^S$, implying that swapping jobs to move towards ascending order of due dates never makes the objective function worse.

1.2.1.5 Conclusion We may perform these sort of swaps using the `bubble sort` algorithm until we terminate at a schedule that is sorted by ascending order of due dates. Due to the proofs above, the $L_{\max}$ of this schedule is necessarily no worse than any other possible schedule. As it is not worse than any other possible schedule, it is necessarily optimal.

2 Question 2

For the sequence dependent setups S of a particular sequence, $C_{max} = \sum_{i=1}^{n} p_i + \sum_{s \in S} s$.

Thus $\sum_{i}^{n} p_i$ is a constant part of the objective function, and the only variables are in S.

We may turn this into a TSP by using s_{jk} as an analogy for distance (which is constant for given j, k). Additionally, we use x_{jk} as binary variables denoting if the pair of jobs j and k are adjacent (1 if they are, 0 if they are not).

As such, the TSP formulation is as follows:

$$\min \sum_{i=1}^{n} p_i + \sum_{j=1}^{n} \sum_{k=1}^{n} s_{jk} x_{jk}$$

s.t.

$$\sum_{j=1}^{n} x_{jk} = 1 \text{ for } j = 1, \ldots, n$$

$$\sum_{k=1}^{n} x_{jk} = 1 \text{ for } k = 1, \ldots, n$$

$$\sum_{j=1}^{|S|} \sum_{k=1}^{|S|} x_{jk} \leq |S| - 1$$

$$x_{jk} \in \{0, 1\} \text{ for } i, j = 1, \ldots, n$$

It is worth noting that as we are free to choose any job as a starting point, that there are a set of n optimal schedules with the same objective value.

3 Question 3

As processing time is 1 for all jobs, C_j is integral from [1, j] and increases in constant increments (in this case, increments of 1), we may see this as an assignment problem. The problem is simply choosing which slot (k) is filled by which job (k) in a one-to-one fashion. We may use binary variables x_{jk} for this purpose.

$$\min \sum_{j=1}^{n} w_j T_j$$

s.t.

$$
\begin{aligned}
T_j &\geq C_j - d_j \text{ for } j = 1, \ldots, n \\
T_j &\geq 0 \text{ for } j = 1, \ldots, n \\
C_j &= \sum_{k=1}^{n} k x_{jk} \text{ for } j = 1, \ldots, n \\
\sum_{j=1}^{n} x_{jk} &= 1 \text{ for } j = 1, \ldots, n \\
\sum_{k=1}^{n} x_{jk} &= 1 \text{ for } k = 1, \ldots, n \\
x_{jk} &\in \{0, 1\} \text{ for } i, j = 1, \ldots, n
\end{aligned}
$$

4 Question 4

4.1 Nonnegative Weights

4.1.1 Constructive Proof

- The unit penalty functions U_j are non-decreasing.
- Their non-decreasing property is maintained while scaling any of them by non-negative weights. This is true of scaling any non-decreasing function with any non-negative constant.
- The sum of non-decreasing functions is itself non-decreasing

$\therefore \sum_{j=1}^{n} w_j U_j(C_j)$ is non-decreasing with increasing C, meaning it is regular.

4.2 Some Negative Weights

It is always nonregular.

4.2.1 Proof by Contradiction

Let γ' be a relaxation of γ to allow some negative weights.

We assume that γ' is regular.

γ' is regular $\implies \gamma'(S_1) > \gamma'(S_2)$ $\forall$ schedules S_1, S_2 where $C^{S_1} \geq C^{S_2}$ and $\exists k$ such that $C_j^{S_1} > C_j^{S_2}$

However, if we consider the case where $C_j^{S_1} = C_j^{S_2} \forall j \in J$ k and $w_j < 0$, we see a contradiction when considering the following:

$$\gamma'(S_1) - \gamma'(S_2) = w_j(C_j^{S_1} - C_j^{S_2})$$

Examining this closely, we note that:

1. $C_j^{S_1} > C_j^{S_2} \implies C_j^{S_1} - C_j^{S_2} > 0$
2. As $w_j < 0$ and $C_j^{S_1} - C_j^{S_2} > 0$, $w_j(C_j^{S_1} - C_j^{S_2}) < 0$
3. $w_j(C_j^{S_1} - C_j^{S_2}) < 0 \implies \gamma'(S_1) - \gamma'(S_2) < 0$, thus violating $\gamma S_1) > \gamma'(S_2)$

This violation is sufficient proof that having a single negative weight makes the objective function irregular. The logic is extensible to demonstrate that any number of negative weights necessarily breaks regularity.

5 Question 5

5.1 Question 5 (1)

It is regular due to the following:

1. Non-negative scalings of non-decreasing functions result in functions that are still non-decreasing (here, scaling by non-negative weights w_1 and w_2)
2. Summations of nondecreasing functions result in a non-decreasing function

Thus the final function is non-decreasing and is therefore regular.

5.2 Question 5 (2)

It is not regular.

Consider a toy univariate case where γ_2 increases far quicker than γ_1. The overall function would no longer be increasing, but instead decreasing.

E.g. in the domain of $C_1 \geq 1$, consider:

$\gamma_1(C_1) := C_1$

$\gamma_2(C_1) := C_1^2$

While these two functions are regular on their own, their ratio decreases in the domain of $x \geq 1$ and thus $\frac{\gamma_1}{\gamma_2}$ is irregular.