University of Waterloo
CO 454 — Scheduling
Spring 2013
Problem Set 1
Siwei Yang - 20258568

1. [Problem 1: LCL rule]

   - (a)

     Applying LCL from the completion time of last job at time $4 + 8 + 12 + 7 + 6 + 9 + 9 = 55$

Jobs	1	2	3	4	5	6	7
$f_j(x)$	165	77	3025	82.5	$70 + \sqrt{55}$	88	77

     picking job 2 with minimum weight at the time
     at time $4 + 12 + 7 + 6 + 9 + 9 = 47$

Jobs	1	3	4	5	6	7
$f_j(x)$	141	2209	70.5	$70 + \sqrt{47}$	75.2	65.8

     picking job 7 with minimum weight at the time
     at time $4 + 12 + 7 + 6 + 9 = 38$

Jobs	1	3	4	5	6
$f_j(x)$	114	1444	57	$70 + \sqrt{38}$	60.8

     picking job 4 with minimum weight at the time
     at time $4 + 12 + 6 + 9 = 31$

Jobs	1	3	5	6
$f_j(x)$	93	961	$70 + \sqrt{31}$	49.6

     picking job 6 with minimum weight at the time
     at time $4 + 12 + 6 = 22$

Jobs	1	3	5
$f_j(x)$	66	484	$70 + \sqrt{22}$

     picking job 1 with minimum weight at the time
     at time $12 + 6 = 18$

Jobs	3	5
$f_j(x)$	324	$70+\sqrt{18}$

picking job 5 with minimum weight at the time

at time $12 = 12$

Jobs	3
$f_j(x)$	144

picking job 3 with minimum weight at the time

thus, the schedule comes out as 3, 5, 1, 6, 4, 7, 2 with objective value 144

- (b)

  At any step of the modified LCL rules, let l' be the job being scheduled. If $l' \notin L(J)$, then there is at least one job l such that $l' \to l$. In the final schedule being produced, we have job l scheduled before l' which violates the prec constraints. Thus, at each step, given l' is the job being scheduled, we have $l' \in L(J)$.

  Assuming the minimum possible f_{max} at each step is v, referring to the proof we have in class, we always have the equation as we proceed with scheduling:

  $$v \geq \min_{k \in L(J)} f_k(\sigma_{j \in J} p_j) \tag{1}$$

  and let J' be any subset of J, we have $v \geq$ the minimum possible f_{max} from scheduling J'.

  Thus, **the modified LCL is making optimal decision at every step which means it's an optimal algorithm for the problem**.

- (c)

  Applying modified LCL from the completion time of last job at time $4+8+12+7+6+9+9 = 55$

Jobs	1	2	3	4	5	6	7
$f_j(x)$	165	77	3025	82.5	$70+\sqrt{55}$	88	77

  where J = 2, 3, 4, 6. Picking job 2 with minimum weight at the time

  at time $4+12+7+6+9+9 = 47$

Jobs	1	3	4	5	6	7
$f_j(x)$	141	2209	70.5	$70 + \sqrt{47}$	75.2	65.8

where J = 3, 4, 6. Picking job 6 with minimum weight at the time at time $4 + 12 + 7 + 6 + 9 = 38$

Jobs	1	3	4	5	7
$f_j(x)$	114	1444	57	$70 + \sqrt{38}$	53.2

where J = 3, 4, 7. Picking job 7 with minimum weight at the time at time $4 + 12 + 7 + 6 = 29$

Jobs	1	3	4	5
$f_j(x)$	87	841	43.5	$70 + \sqrt{29}$

where J = 1, 3, 4. Picking job 4 with minimum weight at the time at time $4 + 12 + 6 = 22$

Jobs	1	3	5
$f_j(x)$	66	484	$70 + \sqrt{22}$

where J = 1, 3, 5. Picking job 1 with minimum weight at the time at time $12 + 6 = 18$

Jobs	3	5
$f_j(x)$	324	$70 + \sqrt{18}$

where J = 3, 5. Picking job 5 with minimum weight at the time at time $12 = 12$

Jobs	3
$f_j(x)$	144

where J = 3. Picking job 3 with minimum weight at the time thus, the schedule comes out as 3, 5, 1,4, 7, 6, 2 with objective value 144

2. [Problem 2: Running time analysis and O(.) notation]

- (a) Driven by the two loops, the inner statements will be executed $\frac{n*(n+1)}{2}$ times. And the three statements cost n + 1 operations on average. Thus, the running time of this algorithm is bounded by $\Theta(n^3)$

- (b)

  for i = 1, 2, ..., n **do**
      initialize MAX, MIN = A[i]
      for j = i, i + 1, ..., n **do**
          Update MAX if A[j] ¿ MAX
          Update MIN if A[j] ¡ MIN
          Set B[i, j] = MAX - MIN.
      end for
  end for

  This algorithm has the same bahavior as the previous one because for every i, j pair, MAX and MIN represents the max and min values from A[i], A[i+1] through A[j]. And B[i, j] is assigned to MAX - MIN for every i, j pair.

  Driven by the two loops, the inner statements will be executed $\frac{n*(n+1)}{2}$ times. And the three statements cost 3 operations. Thus, the running time of this algorithm is bounded by $\Theta(n^2)$

3. [<u>Problem 3</u>: $1|prec|f_{max}$ and EDD rules for $1|r_j, pmtn|L_{max}$ and $1|r_j, pmtn, prec|L_{max}$]

- (a)

  Consider any schedule S for I, it's objective value v is obtained at:

$$v = \max_{j \in J} f_j(C_j) \tag{2}$$

  And, use the same schedule S for I', it's objective value v' is obtained at:

$$v' = \max_{j \in J} f'_j(C_j) \tag{3}$$

  For any pair of jobs J_m, J_n such that $J_m \to J_n$, if $f'_{J_m}(C_{J_m}) \leq f'_{J_n}(C_{J_n})$ then have $J_m \in J'$. Then we easily have:

$$v' = \max_{j \in J} f'_j(C_j) = \max_{j \notin J'} f'_j(C_j) \tag{4}$$

  By observing the definition of f's, we also know that $f'_j = f_j$ for $j \notin J'$ because any j such that $f'_j(t) = f'_{j'}(t + p_{j'})$ where j' is dependent on j will have $f'_j(C_j) = f'_{j'}(C_j + p_{j'}) \leq f'_{j'}(C_{j'})$ which means $j \in J'$. This leads to:

$$v' = \max_{j \notin J'} f_j(C_j) \leq \max_{j \in J} f_j(C_j) = v \tag{5}$$

On the other hand, we have:

$$v = \max_{j \in J} f_j(C_j) \leq \max_{j \in J} f'_j(C_j) = v' \tag{6}$$

since $f_j \leq f'_j$ for all j.
Therefore, v = v'and **S have same object value for I and I'**
And now assume S is a schedule produced for I', let J_m, J_n be any pair of jobs such that $J_m \to J_n$ in I. If $C_{J_m} \geq C_{J_n}$, then we must have:

$$f'_{J_m}(C_{J_m}) \leq f'_{J_n}(C_{J_m}) \tag{7}$$

However, since $f'_{Jm}(t) = \max f'_{J_m}(t), f'_{J_n}(t + p_{J_n})$, $f'_{Jm}(t) > f'_{Jn}(t)$ which means equation 7 is not satisfable. Thus, we have $C_{J_m} < C_{J_n}$. Therefore, S is a schedule satisfy all precedence requirement from I. In other words, **S is a feasible schedule for I**;

- (b)
  Follow the procedure defined in part (a) except the function definition part where we have:

$$d'_j = \min d'_j, d'_k - p_k \tag{8}$$

- (c)
- (d)

4. [Problem 4: $1|r_j; pmtn|\Sigma_j w_j C_j$]

   Consider the following instance where ϵ is a small positive number:

Job	release date	weight	processing time
1	0	1	2
2	1	$2+\epsilon$	2
3	2	$2+\epsilon$	1

The optimal schedule is to process the jobs in the sequence 1, 3, 2 where the end cost comes at:

$$2 * 1 + 3 * (2 + \epsilon) + 5 * (2 + \epsilon) \tag{9}$$

However, the schedule generated by WSRPT heuristic goes as: run job 1 from time 0 to 1; run job 2 to completion from time 1 to 3; run job 3 to completion from time 3 to 4; run job 1 from time 4 to 5 where the end cost comes at:

$$3 * (2 + \epsilon) + 4 * (2 + \epsilon) + 5 * 1 \tag{10}$$