70068 Scheduling and Resource Allocation CW

Ali Abbas

November 26, 2024

1 Question 1

1.1 LCL Proof

The *Least Cost Last* (LCL) rule solves the scheduling problem $1|prec|f_{\max}$ optimally, in $O(n^2)$ time, by constructing an optimal schedule from back to front.

Definitions

- $N = \{1, 2, \ldots, n\}$ is the index set of all jobs
- $L \subseteq N$ is the subset of jobs without successors (which can be scheduled last)
- $p(S) = \sum_{j \in S} p_j$ is the total processing time of a subset S
- $f^*_{\max}(S)$ is the cost of the optimal schedule for subset S
- $f_{\max}$ denotes the maximum (not necessarily optimal) cost across all jobs in a schedule
- $f_j(p(N))$ is a cost function which assigns a penalty/cost depending on job j and the time at which the machine has finished processing, $p(N)$
  - In the notation of the coursework specification $g_j(\cdot) = f_j(\cdot)$
  - In the next section we'll use $g_j(C_j) = T_j = \max(0, C_j - d_j)$ (tardiness)

Proof and Discussion

1. One job in L must be scheduled last. A job can be selected for the final job which minimises $f_j(p(N))$, but it can never result in a cost lower than the cost of the optimal schedule, otherwise that would be the optimal schedule. This is expressed as such:

   $$f^*_{\max}(N) \geq \min_{j \in L} f_j(p(N))$$

2. And removing a job $j \in N$ can't increase the optimal cost any more, since the cost of scheduling a job last is ≥ 0, so omitting it cannot result in a more expensive schedule:

   $$f^*_{\max}(N) \geq f^*_{\max}(N - \{j\}),\ \forall j \in N$$

3. We can then select a $J_l \in L$ that minimises $f_j(p(N))$:

   $$f_l(p(N)) = \min_{j \in L} f_j(p(N))$$

   Which gives us:

   $$f^*_{\max}(N) \geq \max\{f_l(p(N)), f^*_{\max}(N - \{l\})\}$$

4. The right hand side of the above inequality is the cost of an optimal schedule, where J_l is processed last, so you can recursively apply the LCL rule to $N - \{J_l\}$ and construct a schedule in reverse order
   - Since J_l is found in $O(n)$ time, then with n, repeated applications of the LCL rule yields the optimal schedule in $O(n^2)$ time

You can also show that LCL is optimal by using the *adjacent pairwise interchange argument*:

- You take an optimal schedule S and assume it isn't an LCL schedule
- Since it's not an LCL schedule, there must exist at least one pair where a lower cost job precedes a higher cost one

- It can then be shown that swapping these jobs yields a lower cost schedule, which is a contradiction since we took an optimal schedule
- So by virtue of the fact that a lower cost job can precede a higher cost one leads to a contradiction
- And so such pairs cannot exist for an optimal schedule, so we can reject the assumption that this isn't an LCL schedule
- And by contradiction then we have that LCL is optimal for $1|prec|f_{\max}$

Small Example

Example Setup:
Jobs: J1, J2, J3, J4, J5
Processing Times: p=[2,3,1,2,3]
Dude Dates: d=[6,5,7,4,9]
Precedence Constraints (DAG): J1 $\rightarrow$ J2, J1 $\rightarrow$ J3, J4 $\rightarrow$ J5
Cost Function: $f_j(C_j) = T_j = max(0, C_j - d_j)$

Iterations We can schedule jobs in reverse order using the *Least Cost Last* (LCL) rule:

1. Available jobs: $V = \{J_2, J_3, J_5\}$ (no successors)
   $p(N) = 11$
   $$f_2(p(N)) = \max(0, 11-5) = 6, \quad f_3(p(N)) = \max(0, 11-7) = 4, \quad f_5(p(N)) = \max(0, 11-9) = 2$$
   Select J_5 (minimises $f_j(p(N))$)
   Updated schedule: $[J_5]$
2. Available jobs: $V = \{J_2, J_3, J_4\}$
   $p(N) = 8$
   $$f_2(p(N)) = \max(0, 8-5) = 3, \quad f_3(p(N)) = \max(0, 8-7) = 1, \quad f_4(p(N)) = \max(0, 8-4) = 4$$
   Select J_3
   Updated schedule: $[J_3, J_5]$
3. Available jobs: $V = \{J_2, J_4\}$
   $p(N) = 7$
   $$f_2(p(N)) = \max(0, 7-5) = 2, \quad f_4(p(N)) = \max(0, 7-4) = 3$$
   Select J_2
   Updated schedule: $[J_2, J_3, J_5]$
4. Available jobs: $V = \{J_1, J_4\}$
   $p(N) = 4$
   $$f_1(p(N)) = \max(0, 4-5) = 0, \quad f_4(p(N)) = \max(0, 4-4) = 0$$
   Select J_4 (breaking the tie arbitrarily)
   Updated schedule: $[J_4, J_2, J_3, J_5]$
5. Available jobs: $V = \{J_1\}$
   $p(N) = 2$
   $$f_1(p(N)) = \max(0, 2-5) = 0$$
   Select J_1
   Final schedule: $[J_1, J_4, J_2, J_3, J_5]$

Results

- Final schedule: $[J_1, J_4, J_2, J_3, J_5]$
- Completion times:
  $$C_1 = 2,\ C_4 = 4,\ C_2 = 7,\ C_3 = 8,\ C_5 = 11$$
- Tardiness:
  $$T_1 = 0,\ T_4 = 0,\ T_2 = 2,\ T_3 = 1,\ T_5 = 2$$
- Maximum cost:
  $$f_{\max} = \max(T_1, T_2, T_4, T_3, T_5) = 2$$

1.2 LCL Implementation

2 Question 2

2.1 Tabu Search

2.2 Best Schedule