Single machine scheduling with common due date and controllable processing times

Ji-Bo wang (2006)
Contents

1. Introduction
2. Optimal compressions
3. Optimal sequences
4. A special case
5. Conclusions
Introduction

❖ Problem review

• Single machine scheduling
  ✓ Common due date
  ✓ Controllable processing times
    • Normal processing time $t_i$
    • Crash processing time $t'_i$
    • Actual processing $p_i \in [t'_i, t_i]$
  ✓ Compression cost (processing time)
    • Linear compression cost function

• Polynomial solvable problem
  ✓ time complexity $O(n^3)$
  ✓ A special case solvable in $O(n \log n)$

• Literature review
  ✓ Vickson (1980) first paper on controllable processing time scheduling problems.
  ✓ Panwalkar and Rajagopalan (1992) considered the common due date assignment in single M/C with objective is the sum of penalties based on earliness, tardiness and processing time compressions.
  ✓ Cheng et al. (1996) considered a due date assignment and single machine scheduling in which a penalty for due dates is added to the objective function
Introduction

Problem description

- Problem
  - Single machine scheduling with common due date and controllable processing time
- Objective
  - Cost function $f(d, \sigma, \tilde{x}) = \sum_{i=1}^{n} w_i |L_{\sigma(i)}| + w_0d + \sum_{i=1}^{n} G_{\sigma(i)}x_{\sigma(i)}$
- Decision variable
  - Compressions of the processing times
  - Job sequence
- Algorithms
  - Change problem into an assignment problem (Hungarian method)
- Assumption
  - All jobs are available at time zero
  - No preemption and splitting
  - Machine is available at time zero and for whole duration
  - Machine can not process two or more jobs simultaneously
  - After the process has started, no idle time can be inserted in the schedule.
Production and Logistics Information

Introduction

Notations

• $\sigma$ the sequence of jobs to be processed by the machine
• $\sigma(i)$ job in the $i$th position
• $t_i$ the normal processing time of job $j_i$
• $t'_i$ the crash processing time of job $j_i$
• $G_i$ the per time compression cost of job $j_i$
• $m_i$ the maximum reduction in processing time of job $j_i$, $m_i = t_i - t'_i$.
• $x_j$ amount of compression of job $j$, $0 \leq x_j \leq m_j$
• $\bar{x}$ the vector job compressions $ar{x} = (x_1, x_2, ..., x_n)$
• $p_i$ the actual processing time of job $j_i$
• $C_i$ completion time of job $j_i$
• $d$ the common due date
• $w_i$ the weights do not correspond with the jobs but with the position
• $|L_{\sigma(i)}|$ the absolute value in lateness, $|L_{\sigma(i)}| = |C_{\sigma(i)} - d|$
Introduction

❖ Objection function

- Cost functions:

\[
 f(d, \sigma, \tilde{x}) = \sum_{i=1}^{n} w_i |L_{\sigma(i)}| + w_0 d + \sum_{i=1}^{n} G_{\sigma(i)} x_{\sigma(i)} 
\]

- Simplification:
  - Introduce a dummy job 0 with actual processing time \( p_0 = 0 \) and weight \( w_0 \).

\[
 f'(d, \sigma, \tilde{x}) = \sum_{i=0}^{n} w_i |L_{\sigma(i)}| + \sum_{i=1}^{n} |G_{\sigma(i)} x_{\sigma(i)}| 
\]

\[
w_0 d = w_0 |L_{\sigma(i)}|
\]
Optimal compressions

✓ Property

- The following properties using to formulate the problem as an assignment problem

✓ Property 1

For any given $\tilde{x}$, there exists an optimal sequence $\sigma^*$ without any machine idle time between the processing of jobs, and first job starts at time zero

✓ proof

Case 1: $d < t$ move job $j$ and the jobs scheduled after $j'$ by an amount of $\Delta = s - t$ units without increasing the objective value.

case 2: $d > s$ move job $i$ and the jobs scheduled before $i$ by the same amount of units to the right without increasing the objective value.

case 3: $t < d < s$ move $i$ with its predecessors to the right and $j$ with its successors to the left such that $i$ finishes at time $d$ and $j$ starts at time $d$ without increasing the objective value.
Optimal compressions

Property

✓ Property 2

Given \( \tilde{x} \), there exists an optimal sequence \( \sigma^* \), in which the kth job is completed at \( d \), i.e. \( d = \sum_{i=0}^{k} p_i \), where \( k \) meet

\[
\sum_{j=0}^{k-1} w_j \leq \sum_{j=k}^{n} w_j \quad \text{and} \quad \sum_{j=0}^{k} w_j \geq \sum_{j=k+1}^{n} w_j.
\]

✓ Proof

First proof an optimal solution \( d_{opt} \) is the finishing time of some jobs.

Consider a solution \( \sigma, d \) with \( C_{\sigma(i)} < d < C_{\sigma(i+1)} \) and let \( Z \) be the corresponding objective value. Define:

\( x := d - C_{\sigma(i)} \) and \( y := C_{\sigma(i+1)} - d \)

Let \( Z' \) and \( Z'' \) be the objective value for \( d = C_{\sigma(i)} \) and \( d = C_{\sigma(i+1)} \):

\[
Z' = Z + x \sum_{j=i+1}^{n} w_j - x \sum_{j=0}^{i} w_j = Z + x(\sum_{j=i+1}^{n} w_j - \sum_{j=0}^{i} w_j)
\]

\[
Z'' = Z - y \sum_{j=i+1}^{n} w_j + y \sum_{j=0}^{i} w_j = Z - y(\sum_{j=i+1}^{n} w_j - \sum_{j=0}^{i} w_j)
\]

\( Z' \leq Z \) if \( \sum_{j=i+1}^{n} w_j - \sum_{j=0}^{i} w_j \leq 0 \) and \( Z'' \leq Z \) otherwise
Optimal compressions

❖ Property

✓ Proof (cont’)

assume that \( d_{\text{opt}} = C_{\sigma(k)} \), where \( \sigma \) is an optimal sequence.

\[
Z' = Z + x \sum_{j=i+1}^{n} w_j - x \sum_{j=0}^{i} w_j = Z + x \left( \sum_{j=i+1}^{n} w_j - \sum_{j=0}^{i} w_j \right)
\]

Opt. positive

\[
Z'' = Z - y \sum_{j=i+1}^{n} w_j + y \sum_{j=0}^{i} w_j = Z - y \left( \sum_{j=i+1}^{n} w_j - \sum_{j=0}^{i} w_j \right)
\]

Opt. negative

\[
\sum_{j=0}^{k-1} w_j \leq \sum_{j=k}^{n} w_j
\]

\[
\sum_{j=0}^{k} w_j \geq \sum_{j=k+1}^{n} w_j
\]
Optimal compressions

**Property**

**✓ Property 3**

*Given $\bar{x}$, assuming that the waiting time of one job coincides with the common flow allowance the objective function can be rewritten as*

$$f(d, \sigma, \bar{x}) = \sum_{i=0}^{n} w_i |L_{\sigma(i)}| + \sum_{i=1}^{n} G_{\sigma(i)} x_{\sigma(i)}$$

Where $\lambda_j = \begin{cases} \sum_{v=0}^{j-1} w_v & \text{for } j = 1, 2, \ldots, k, \\ \sum_{v=j}^{n} w_v & \text{for } j = k + 1, k + 2, \ldots, n \end{cases}$ is the *positional weight* which arises if a job occupies the $j$th position in a schedule.

**✓ Proof**

\[
\sum_{j=0}^{k} w_j(C_{\sigma(k)} - C_{\sigma(j)}) + \sum_{j=k+1}^{n} w_j(C_{\sigma(j)} - C_{\sigma(k)}) = \sum_{j=0}^{k} w_j \sum_{\nu=1}^{k} p_{\sigma(\nu)}(\sum_{j=0}^{j-1} w_j) + \sum_{j=k+1}^{n} p_{\sigma(\nu)}(\sum_{j=0}^{n} w_j).
\]

\[
= \sum_{j=1}^{n} p_{\sigma(j)} \lambda_j
\]
Optimal compressions

**Property**

✓ **Property 4**

*For the controllable processing times problems with linear compression costs. There exists an optimal schedule such that there no partially compressed jobs.*

✓ **Proof**

Using properties 2 and 3, and substituting $C_{\sigma(j)} = \sum_{i=1}^{j} p_{\sigma(i)}$, $x_{\sigma(j)} = t_{\sigma(j)} - p_{\sigma(j)}$ and $d = \sum_{i=0}^{k} p_i$ into objective function, have

$$f(d, \sigma, \tilde{x}) = \sum_{i=1}^{n} w_i |L_{\sigma(i)}| + w_0 d + \sum_{i=1}^{n} G_{\sigma(i)} x_{\sigma(i)}$$

where $\lambda_j$ is *positional weight*,

$$\lambda_j = \begin{cases} \sum_{v=0}^{j-1} w_v - G_{\sigma(j)} & \text{for } j = 1, 2, \ldots, k, \\ \sum_{v=j}^{n} w_v - G_{\sigma(j)} & \text{for } j = k + 1, k + 2, \ldots, n, \end{cases}$$

for the optimal sequencing, the optimal processing time:

$$P^*_{\sigma(j)} = \begin{cases} t_{\sigma(j)}, & \text{if } \lambda_j < 0, \text{ Negative position weight should be } t_{\sigma(j)} \\ p'_{\sigma(j)}, & \text{if } \lambda_j = 0, \text{ Value 0} \\ t'_{\sigma(j)}, & \text{if } \lambda_j > 0, \text{ Positive position weight should be } t'_{\sigma(j)} \end{cases}$$
Optimal sequences

Optimal processing times and compressions can be computed for any given sequence, the problem reduces to a pure sequencing problem.

Assignment problem

Where, \( t'_j \leq p'_j \leq t_j \), \( x_{ij} = 1 \) if job \( j \) is scheduled in position \( j \), and \( x_{ij} = 0 \), otherwise.

\[
\begin{align*}
\min & \quad \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_{ij} p_{ij} x_{ij} \\
\text{subject to} & \quad \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n, \\
& \quad \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n, \\
& \quad x_{ij} = 0 \text{ or } 1, \quad i, j = 1, 2, \ldots, n.
\end{align*}
\]

\[
\lambda_{ij} = \begin{cases} 
\sum_{v=0}^{i-1} w_v - G_j & \text{for } i = 1, 2, \ldots, k, \\
\sum_{v=i}^{n} w_v - G_j & \text{for } i = k + 1, k + 2, \ldots, n
\end{cases}
\]

\[
p_{ij} = \begin{cases} 
t_j, & \text{if } \lambda_{ij} < 0, \\
p'_j, & \text{if } \lambda_{ij} = 0, \\
t'_j, & \text{if } \lambda_{ij} > 0
\end{cases}
\]

Solving an assignment problem of size \( n \) requires an effort of \( O(n^3) \) using Hungarian method.
A special case

General case time complexity is $O(n^3)$. Consider a special case $G_i = G$ and $t_i - t'_i = m$, $i = 1, 2, \ldots, n$, which can be solved in $O(n \log n)$ time.

✓ Theorem 1

A special case case $G_i = G$ and $t_i - t'_i = m$, $i = 1, 2, \ldots, n$, the optimal sequence $\sigma^*$ is the sequence obtained from matching the position weights in descending order with the normal processing time $s$ in ascending order.

✓ Proof

Case: $\lambda_{\sigma(j)} < 0$ and $\lambda_{\sigma(j)} > 0$

\[
\begin{align*}
  f(\sigma', x_i) - f(\sigma^*, x_i) &= \hat{\lambda}_{\sigma(j)} t_{\sigma(j+1)} + \hat{\lambda}_{\sigma(j+1)} t'_{\sigma(j)} - (\hat{\lambda}_{\sigma(j)} t_{\sigma(j)} + \hat{\lambda}_{\sigma(j+1)} t'_{\sigma(j+1)}) \\
  &= \hat{\lambda}_{\sigma(j)} t_{\sigma(j+1)} + \hat{\lambda}_{\sigma(j+1)} (t_{\sigma(j)} - m) - \hat{\lambda}_{\sigma(j)} t_{\sigma(j)} \\
  &\quad - \hat{\lambda}_{\sigma(j+1)} (t_{\sigma(j+1)} - m) \\
  &= (t_{\sigma(j+1)} - t_{\sigma(j)}) (\hat{\lambda}_{\sigma(j)} - \hat{\lambda}_{\sigma(j+1)}) < 0.
\end{align*}
\]

Other case can be solved in a similar manner.

In a optimal sequence $\sigma^*$, two adjacent positions $j$ and $j+1$ such that $\lambda_{\sigma(j)} < \lambda_{\sigma(j+1)}$ but $t_{\sigma(j)} < t_{\sigma(j+1)}$, interchanging the two jobs obtain sequence $\sigma'$. 
A special case

A $O(n \log n)$ algorithm for the special case.

✓ Algorithm 1

Step 1. Weight $n$ positions by using Eq. (3) for the model (1).
Step 2. Rank the position weights $\lambda_i$ for the model (1) in descending order of magnitude such that the largest $\lambda_i$ is ranked 1 and smallest $\lambda_i$ is ranked $n$. Break ties arbitrarily.
Step 3. Find the optimal sequence by matching the position weights in descending order with the jobs in ascending order of their normal processing times.
Step 4. Calculate the optimal processing times by using Eq. (4) for the model (1).
Step 5. Calculate the optimal compressions by using Eq. (5) for the model (1).

✓ Example:

\[ \lambda_{\sigma(i)} = \begin{bmatrix} 8 & 6 & 4 & 2 & 0 & 1 & 3 & 5 & 7 \end{bmatrix} \]

\[
\begin{array}{cccccccc}
\text{Rank} & 8 & 1 & 3 & 5 & 7 & 1 & 6 & 4 & 2 \\
\end{array}
\]

Job with largest normal processing time

Job with smallest normal processing time

Step 2 with a time complexity $O(n \log n)$, and step 1, 4 and 5 can be completed in $O(n)$. 

Conclusion

- **Single machine scheduling problem**
  - Common due date assignment cost
  - Processing time compression cost
  - Position weight

- **Suggested**
  - Modelled as an assignment problem $O(n^3)$
  - A special case in suggested algorithm $O(n \log n)$

- **Adv & Disadv**
THANK YOU