Machine Scheduling Problem (MSP)

9. Example of Multi-Objective MSP

The object can describe as a set of n jobs $N = \{1,2,\dots,n\}$ on a single machine to find $\sigma \in S$ (where S is the set of all feasible schedules) so they can be us full to specify whether that minimize the multi-criteria problem.

9.1 Formulation of $1//(\sum C_j, \sum T_j)$ Problem

Our objective is to find efficient schedule σ∈S (where S is the set of all feasible schedule) that minimizes the multi-criteria $(\sum C_j, \sum T_j)$ for the $1//(\sum C_j, \sum T_j)$ problem. This problem belongs to simultaneous optimization and can be written as:

$(\sum C_j, \sum T_j)$. The $1//(\sum C_j, \sum T_j)$ problem can be written as:

$$\left.\begin{array}{ll} \multicolumn{2}{c}{Min\{\sum C_j, \sum T_j\}} \\ \multicolumn{2}{c}{\text{subject to}} \\ C_j \ge p_{\sigma(j)}, & j = 1,2,\dots,n. \\ C_j = C_{j-1} + p_{\sigma(j)}, & j = 2,3,\dots,n. \\ T_j \ge C_j - d_{\sigma(j)}, & j = 1,2,\dots,n. \\ T_j \ge 0, & j = 1,2,\dots,n. \end{array}\right\} \qquad (P)$$

For P-problem, we can deduce subproblem: The $1//\sum C_j + \sum T_j$ Problem:

$$\left.\begin{array}{ll} \multicolumn{2}{c}{Min\{\sum C_j + \sum T_j\}} \\ \multicolumn{2}{c}{\text{subject to}} \\ C_j \ge p_{\sigma(j)}, & j = 1,2,\dots,n. \\ C_j = C_{j-1} + p_{\sigma(j)}, & j = 2,3,\dots,n. \\ T_j \ge C_j - d_{\sigma(j)}, & j = 1,2,\dots,n. \\ T_j \ge 0, & j = 1,2,\dots,n. \end{array}\right\} \qquad (P_1)$$

The goal for the P_1- problem is to find suitable sequence of the jobs on a single machine to reduce the total of completion time and total of tardiness jobs, which is considered just a single object.

<u>Example (7)</u>:

Lets have the following MSP with $n = 4$.

	1	2	3	4

p_j	10	5	9	2
d_j	13	28	24	29

For the following data we have the following results:

i	Sequence	Efficient	Single
1	4,2,3,1	(51,13)	64
2	4,2,1,3	(52,6)	58
3	4,1,2,3	(57,2)	59
4	4,1,3,2	(61,0)	61

Notice that there are 4 efficient solutions for problem (P), while there is one optimal solution for problem (P_1) (2^{nd} solution) not SPT or EDD rule.

9.2 Special Cases for the Problems (P) and (P_1)

For multi-criteria, if the objectives can be optimized individually, then we can deduce that the set of efficient solutions have no more elements only one with extreme values of the individul objective functions. The above fact can be seen in the following special cases:

Case (1): A schedule σ obtained by ordering the jobs in a non-decreasing order of thier processing times (**SPT-rule**) is optimal for both problems if $d_{\sigma(j)} \leq C_{\sigma(j)}$ for all $j = 1,2,\dots,n$.

<u>Example (8)</u>:

Lets have the following data with $n = 4$:

	1	2	3	4
p_j	7	8	10	4
d_j	10	14	22	4

Lets arranged the sequence in SPT rule, we obtain:

	4	1	2	3
p_j	4	7	8	10
d_j	4	10	14	22
C_j	4	11	19	29

Notice that $d_{\sigma(j)} \leq C_{\sigma(j)}$ for all jobs, then the SPT rule gives a unique efficient solution to problem (P) and the optimal solution for problem (P_1), for the sequence (4,1,2,3) with objective value (63,13) and optimal solution 64.

Case (2): From Emmon's theorem, if the **SPT and EDD rules** are identical then there exist **only one** effeceint solution for (P) and (P_1) (see example (8)).

Case (3): If $p_j = p, \forall j$, p is positive integer and a schedule σ obtained by ordering the jobs in a non-decreasing order of due dates (**EDD-rule**) is optimal for both problems. (**prove**).

Case (4): If $d_j = d, \forall j$ is positive integer and a schedule σ obtained by ordering the jobs in a non-decreasing order of processing times (**SPT-rule**) is optimal for both problems. (**prove**).

9. Applying of Solving Methods for Multi-Criteria Function

9.1 Applying of BAB Method

Example (9): Lets have the following MSP with n=4:

	1	2	3	4
p_j	7	8	10	4
d_j	20	14	22	9

We want to solve the problem $\left(\sum C_j, \sum T_j\right)$.

Lets assume that the UB is depending on standard scheduling (j=1,2,3,4).

So that UB=(76,24).

As we know that the LB=sequence part + unsequenced part.

For the unsequenced part we suggest using SPT rule, so we have:

First Level:

1. **For node 1**: apply SPT rule for the unsequence part, we obtain 4-2-3, so the LB1=(66,14); since LB1 not dominated by UB, then node1 is chosen.
2. **For node 2**: apply SPT rule for the unsequence part, we obtain 4-1-3, so the LB2=(68,10); since LB2 not dominated by UB, then node2 is chosen.
3. **For node 3**: apply SPT rule for the unsequence part, we obtain 4-1-2, so the LB3=(74,21); since LB3 not dominated by UB, then node3 is chosen.
4. **For node 4**: apply SPT rule for the unsequence part, we obtain 1-2-3, so the LB4=(63,12); since LB3 not dominated by UB, then node4 is chosen.

From the set of efficient solution S1={(66,14), (68,10), (74,21),(63,12)}we filter the set to obtain new set S1={(68,10),(63,12)}, which are LB for nodes 2 and 4.

Second Level:

1. **For node 2-1**: apply SPT rule for the unsequence part, we obtain 4-3, so the LB21=(77,17); since LB21 not dominated by UB, then node2-1 is chosen.
2. **For node 2-3**: apply SPT rule for the unsequence part, we obtain 4-1, so the LB23=(77,22); since LB23 not dominated by UB, then node2-3 is chosen.

3. **For node 2-4**: apply SPT rule for the unsequence part, we obtain 1-3, so the LB24=(68,10); since LB24 not dominated by UB, then node2-4 is chosen.
4. **For node 4-1**: apply SPT rule for the unsequence part, we obtain 2-3, so the LB41=(63,12); since LB41 not dominated by UB, then node4-1 is chosen.
5. **For node 4-2**: apply SPT rule for the unsequence part, we obtain 1-3, so the LB42=(64,7); since LB42 not dominated by UB, then node4 is chosen.
6. **For node 4-3**: apply SPT rule for the unsequence part, we obtain 1-2, so the LB43=(68,16); since LB43 not dominated by UB, then node4-3 is chosen.

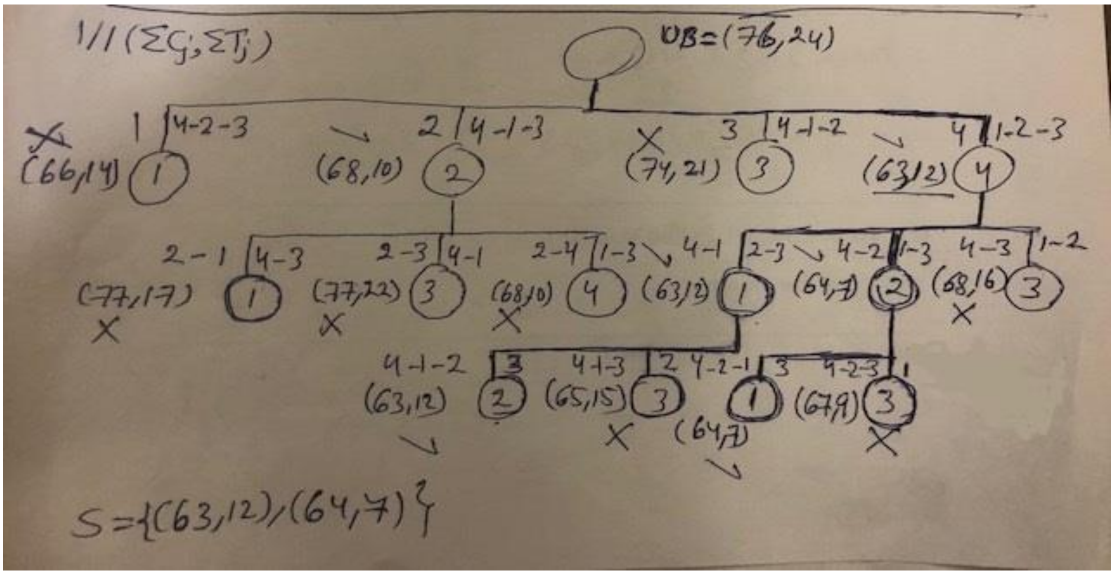
From the set of efficient solution S2={(77,17), (77,22), (68,10),(63,12),(64,7),(68,16)}we filter the set to obtain new set S2={(63,12),(64,7)}.

Third Level:

1. **For node 4-1-2**: remain 3, so the LB412=(63,12); since LB412 not dominated by UB, then node4-1-2 is chosen.
2. **For node 4-1-3**: remain 2, so the LB413=(65,15); since LB413 not dominated by UB, then node4-1-3 is chosen.
3. **For node 4-2-1**: remain 3, so the LB421=(64,7); since LB421 not dominated by UB, then node4-2-1 is chosen.
4. **For node 4-2-3**: remain 1, so the LB423=(67,9); since LB423 not dominated by UB, then node4-2-3 is chosen.

From the set of efficient solution S3={(63,12),(65,15),(64,7),(67,9)}we filter the set to obtain new set S3={(63,12),(64,7)}.

Now we will stop and get the last set of efficient solutions **S={(63,12),(64,7)}**, which obtained from sequences **[4,1,2,3]** and **[4,2,1,3]** respectively.



H.W: Applying BAB with DR for example (9) to solve the problem $\left(\sum C_j, \sum T_j\right)$.

9.2 Heuristic Method for (P) problems

9.2.1 SPT-EDD-SCST(F) Method

The heuristic methods for P problems are considered as a development of the heuristic methods for $1//\sum T_j$ problem.

The first heuristic method depends on SPT and EDD. Since the SPT rule solving the $1//\sum C_j$ problem in polynomial time, so we suggest to improve the heuristic method EDD-ST by order the jobs by SPT rule firstly, and then calculate the objective function, and then put the second job in first place and the other jobs still arranged by SPT rule and calculate the objective function, and so on until obtain n sequences. The process repeated for EDD rule for P-problem (so we called it SPT-EDD-SCST(F)). The algorithm of SPT-EDD-SCST(F) is as follows:

SPT_EDD_SCST(F) Heuristic Algorithm

Step(1): INPUT n, p_j and $d_j, j = 1,2,3,\dots,n,\ \delta = \phi$.

Step(2): Arrange jobs in SPT rule (σ_1), and calculate $F_{11}(\sigma_1) = \left(\sum C_j(\sigma_1), \sum T_j(\sigma_1)\right)$;

$\delta = \delta \cup \{F_{11}(\sigma_1)\}$.

Step(3): FOR $i = 2,\dots,n$, put job i in the first position of σ_{i-1} to obtain σ_i and calculate $F_{1i}(\sigma_i) = \left(\sum C_j(\sigma_i), \sum T_j(\sigma_i)\right)$;

$\delta = \delta \cup \{F_{1i}(\sigma_i)\}$.

ENDFOR;

Step(4): Arrange jobs in EDD rule (π_1), calculate $F_{21}(\pi_1) = \left(\sum C_j(\pi_1), \sum T_j(\pi_1)\right)$;

$\delta = \delta \cup \{F_{21}(\pi_1)\}$.

Step (5): FOR $i=2,\ldots,n$, put job i in the first position of π_{i-1} to obtain π_i and calculate $F_{2i}(\pi_i) = \left(\sum C_j(\pi_i), \sum T_j(\pi_i)\right)$;
$\delta = \delta \cup \{F_{2i}(\pi_i)\}$.
ENDFOR;
Step(6): Filter set δ to obtain as a set of efficient solution of P-problem
Step(7): OUTPUT The set of efficient solutions δ.
Step(8): END.

Example (10): Call example (9) for n=4:
We obtain the following results:

	Ob. Fun.	Seq.	After Filter	Seq
1	(63,12)	4,1,2,3	(63,12) (64,7)	[4,1,2,3] [4,2,1,3]
2	(66,14)	1,4,2,3		
3	(68,10)	2,4,1,3		
4	(74,21)	3,4,1,2		
5	(64,7)	4,2,1,3		
6	(68,10)	2,4,1,3		
7	(66,14)	1,4,2,3		
8	(75,22)	3,4,2,1		

9.2.2 DR-SCST(F) Method

The idea of the second heuristic method is dependent on the heuristic DR-ST which is described in subsection Heuristic Methods for $1//\sum T_j$. The suggested method will be improved to applied on P-problem (so we called it DR-SCST(F)). The algorithm of DR-SCST(F) is as follows:

DR_SCST(F) Heuristic Algorithm

Step(1): INPUT: n, p_j and $d_j, j = 1,2,\ldots,n$.
Step(2): Apply remark(1) to find DR and adjacency matrix A;
$\sigma = \emptyset, N = \{1,2,\ldots,n\}. \delta = \emptyset$.
Step(3): Sort the jobs with minimum p_j which is not contradiction with matrix A say σ_1, if ∃ more than one job break tie arbitrary, $\delta = \delta \cup \{\sigma_1\}$.
Step(4): Sort the jobs with minimum d_j which is not contradiction with matrix A say σ_2, if ∃ more than one job break tie arbitrary, $\delta = \delta \cup \{\sigma_2\}$.
Step(5): Find the dominated sequence set δ' from δ.
Step(6): Calculate $F(\delta')$.
Step(7): OUTPUT The set of efficient solutions δ'.
Step(8): END.

Example (11): Call example (9) for n=4:
We obtain the following $A(G)$:

$$A(G) = \begin{bmatrix} 0 & a_{12} & 1 & 0 \\ a_{21} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

We obtain the following results:

	Obj. Fun.	Seq
1	(63,12)	4,1,2,3
2	(64,7)	4,2,1,3