Machine Scheduling Problem (MSP)

7. Dominance Rule

Reducing the current sequence may be done by using several Dominance Rules (DR's). DR's usually specify some (all) parts of the path to obtain good value for objective function so they can be useful to determine whether a node in BAB method can be ignored before its lower bound (LB) is calculated. Clearly, DR's are particularly useful when a node can be ignored although it has a LB that is less than the optimum solution. The DR's are also useful within the BAB method to cut all nodes that are dominated by others. These improvements lead to very large decrease in the number of nodes to obtain the optimal solution**.**

Definition [**29**]**:** If *G* is a graph that has *n* vertices, then the **matrix** $\boldsymbol{A(G)=[a_{ij}]}$, whose $i^{th}$and j^{th} element is 1 if there is at least one edge between vertex V_i and vertex V_j and zero otherwise, is called the

$$i = j \text{ or } j \nrightarrow i$$

adjacency matrix of *G*, where:

$$a_{ij} = \begin{cases} 1, & \text{if } i \to j. \\ 0, & \text{if } i = j \text{ or } j \nrightarrow i. \\ a_{ij} \text{ and } \bar{a}_{ij}, & i \leftrightarrow j. \end{cases}$$

s.t. the adjacency matrix A(G) is as follows:

$$\boldsymbol{A(G)}=\begin{bmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ \bar{a}_{21} & 0 & a_{23} & \cdots & a_{2n} \\ \bar{a}_{31} & \bar{a}_{32} & 0 & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \bar{a}_{n1} & \bar{a}_{n2} & \bar{a}_{n3} & \cdots & 0 \end{bmatrix}$$

Emmon's Theorem (1) [20]: For the $1/ /\Sigma T_i$ problem, if $p_i \le p_j$ and $d_i \le d_j$ then there exists an optimal sequencing in which job i sequencing before job j.

Al-Magraby's Lemma (1) [20]: For the $1/ /\Sigma T_i$ problem, if $d_j \ge \sum_{i=1}^{n} p_i$, then there exists an optimal sequence in which job j sequencing last.

Remark (1) [7]: For $1//E_{max}$ problem if $p_i \le p_j$ and $s_i \le s_j$, then there exists an efficient solution in which job *i* is sequenced before job *j*.

8. Applying of Solving Methods for Single objective Function

8.1 Applying of BAB Method

8.1.1 Applying of BAB Method without DR

Example (1):

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 1/ /$\sum T_j$.

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

So that UB=24.

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

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

1. **For node 1**: apply EDD rule for the unsequence part, we obtain 4-2-3, so the LB1=14; since LB1=14≤UB=24, then we branch from node 1.

1.1 Now we branch from nodes 2 or 3 or 4; form node 2 we have seq. part 1-2 and usequenced part 4-3 (after applying EDD), so the LB12=18; since LB12=18≤UB=24, then we branch from node 2.

1.1.1 Now we branch from nodes 3 or 4; from node 3 we have seq. part 1-2-3 and unsequence part 4; so the LB123=24.

1.1.2 Now we branch from node 3, so we have the seq. part 1-2-4 and unsequence part 3, so the LB124=18. Since we arrived at the root of the tree and LB124=18≤UB=24, then we change the UB to obtain new **UB=18**; then we apply the backtracking part of BAB for one level.

1.2 Now we branch from node 3; so the seq part is 1-3 and unsequence part is 4-2 (applying EDD), then then LB13=27≥UB=18, so we ignore this node.

1.3 Now we branch from node 4; so the seq part is 1-4 and unsequence part is 2-3 (applying EDD), then then LB14=14≤UB=18, so we branch from this node.

1.3.1 Now we branch from nodes 2 or 3; from node 2 we have seq. part 1-4-2 and unsequence part 3; so the LB142=14, since we get the root then the new **UB=14**;

1.3.2 Now we branch from node 3, so we have the seq. part 1-4-3 and unsequence part 2, so the LB143=17≥UB=14, so we ignore this node. Now we will back track to level 1 (why?).

2. For node 2: apply EDD rule for the unsequence part, we obtain 4-1-3, so the LB2=10; since LB2=10≤UB=14, then we branch from node 2.

2.1 Now we branch from nodes 1 or 3 or 4; form node 1 we have seq. part 2-1 and usequenced part 4-3 (after applying EDD), so the LB21=17; since LB21=17≥UB=14, so we ignore this node.

2.2 form node 3 we have seq. part 2-3 and usequenced part 4-1 (after applying EDD), so the LB23=22; since LB23=22≥UB=14, so we ignore this node.

2.3 form node 4 we have seq. part 2-4 and usequenced part 1-3 (after applying EDD), so the LB24=10; since LB24=10≤UB=14, so we branch from this node.

2.3.1 Now we branch from nodes 1 or 3; from node 1 we have seq. part 2-4-1 and unsequence part 3; so the LB241=10, since we get the root then the new **UB=10**;

2.3.2 Now we branch from node 3, so we have the seq. part 2-4-3 and unsequence part 1, so the LB243=12≥UB=10, so we ignore this node. Now we will back track to level 1 (why?).

3. For node 3: apply EDD rule for the unsequence part, we obtain 4-2-1, so the LB2=22; since LB3=22≥UB=10, so we ignore this node.

4. For node 4: apply EDD rule for the unsequence part, we obtain 2-1-3, so the LB4=7; since LB4=7≤UB=10, then we branch from node 4.

4.1 Now we branch from nodes 1 or 2 or 3; form node 1 we have seq. part 4-1 and usequenced part 2-3 (after applying EDD), so the LB41=12; since LB41=12≥UB=10, so we ignore this node.

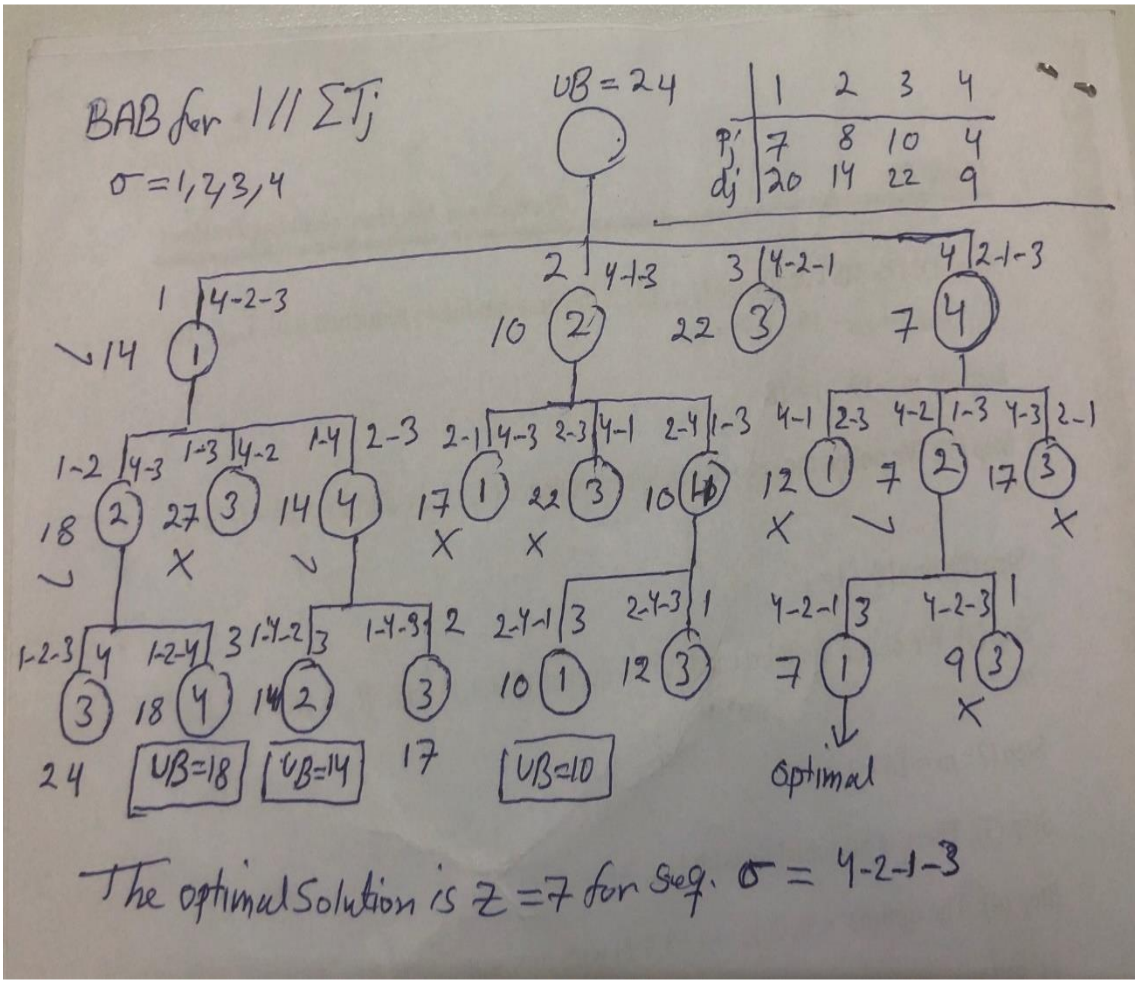
4.2 form node 2 we have seq. part 4-2 and usequenced part 1-3 (after applying EDD), so the LB42=7; since LB42=7≤UB=10, so we branch from this node.

4.2.1 Now we branch from nodes 1 or 3; from node 1 we have seq. part 4-2-1 and unsequence part 3; so the LB421=7, since we get the root then the new **UB=7**;

4.2.2 Now we branch from node 3, so we have the seq. part 4-2-3 and unsequence part 1, so the LB423=9≥UB=7, so we ignore this node. Now we will back track one level.

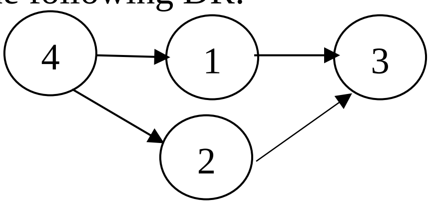
4.3 form node 3 we have seq. part 4-3 and usequenced part 2-1 (after applying EDD), so the LB43=17; since LB43=17≥UB=7, so we ignore this node. now stop since we finish all nodes.

Then the optimal solution is **Z=7**; for the sequence **4-2-1-3**.



8.1.2 Applying of BAB Method with DR

<u>**Example (2)**</u>**:** For the same data in example (1) and applying Emmon's Theorem, we obtain the following DR:



Lets assume that the UB =24.

For the unsequenced part we suggest using EDD rule first we have to start with node which exist in the beginning of the above diagram when using DR, so we have to start as follows:

1. **For node 4:** apply EDD rule for the unsequence part, we obtain 2-1-3, so the LB4=7; since LB4=7≤UB=10, then we branch from node 4.

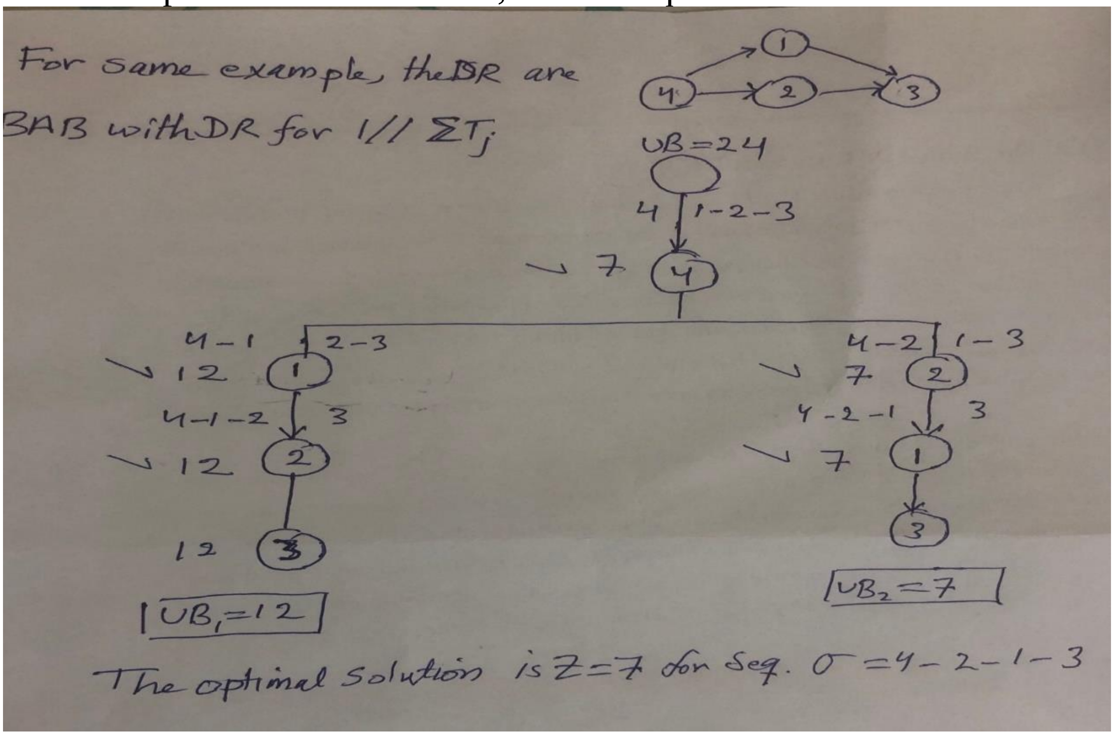
1.1 Now we branch from nodes 1 or 2; form node 1 we have seq. part 4-1 and usequenced part 2-3 (after applying EDD), so the LB41=12; since LB41=12≤UB=24, branch from this node.

1.1.1 Now we branch from nodes 2; from node 2 we have seq. part 4-1-2 and unsequence part 3; so the LB412=12, since we get the root then the new **UB=12**. Now we will back tracking.

1.2 form node 2 we have seq. part 4-2 and usequenced part 1-3 (after applying EDD), so the LB42=7; since LB42=7≤UB=12, so we branch from this node.

1.2.1 Now we branch from nodes 1; from node 1 we have seq. part 4-2-1 and unsequence part 3; so the LB421=7, since we get the root then the new **UB=7**; now stop since we finish all nodes.

Then the optimal solution is **Z=7**; for the sequence **4-2-1-3**.



8.2 Applying of Heuristic Methods

8.2.1 H1: The problem $1//\sum T_j$ considered is NP-hard, so two heuristic or approximate methods are proposed for solving this problem to obtain good solutions. The first suggested heuristic method is using EDD rule for the Sum of Tardiness (which we called it EDD-ST). The idea of this method we start to arrange the jobs by EDD rules and calculate the objective function, and then put the 2nd job in 1st position and then arrange the other jobs by EDD rules and calculate the objective function, we continue this process until obtain n feasible sequences are obtained. The algorithm of EDD-ST is as follows:

EDD_ST Heuristic Algorithm

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

Step(2): Arrange jobs in EDD rule (γ_1), and calculate $F_1(\gamma_1) = \sum T_j(\gamma_1)$, let $\gamma = \gamma_i$

Step(3): FOR $i = 2, \dots, n$, job i in the first position of $\gamma_{i-1}$to obtain γ_i, then calculate $F_i(\gamma_i)$.

Step(4): IF $F_i(\gamma_i) \leq F(\gamma)$ **THEN** $\gamma = \gamma_i$.

ELSE GOTO Step(3).
ENDIF.

Step(5): OUTPUT: The optioned of sequence γ with $F(\gamma)$ value.

Step(6): END.

The second method depends using DR of Sum Tardiness (DR-ST).

Example (3): Use EDD-ST algorithm for Example (1): we obtain

1. 4 , 2 , 1 , 3 $\rightarrow$ 7
2. 2 , 4 , 1 , 3 $\rightarrow$ 10
3. 1 , 4 , 2 , 3 $\rightarrow$ 14
4. 3 , 4 , 2 , 1 $\rightarrow$ 22

Best Seq is $\sum T_j$ = 4 , 2 , 1 , 3 $\rightarrow$ 7 .

8.2.2 H2: The DR-ST (DR of $(1//\sum T_j)$) method is summarized by finding a sequence arranged with minimum p_j and d_j and which it's not contradicted with DR of the problem and then calculate the objective function. The algorithm of DR-ST is stated as follows:

DR_ST Heuristic Algorithm

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

Step(2): Apply Theorem (1) to find adjacency matrix A of DR, $N = \{1,2, \dots, n\}$.

Step(3): FOR $i = 1,2, \dots, n$, find a sequence σ with minimum d_j which is not contradiction with matrix A, if $\exists$ more than one job with same p_j or d_j (or both) then break tie arbitrary.

Step(4): IF $F_i(\sigma_i) \leq F(\sigma)$, **THEN** $\sigma = \sigma_i$

ELSE GOTO Step(3)
ENDIF.

Step(5): OUTPUT: The obtained sequence σ with $F(\sigma)$ value.

Step(6): END.

Example (4): Use DR-ST algorithm for Example (1): we obtain:

1. Node 4: has the min. d_j=9 and not contradiction with A.

2. Node 1 and node 2 are not contradiction with A but node 2 has min. d_j=14.
3. The last node is node 3.

Then the best seq is $\sum T_j$ = 4 , 2 , 1 , 3 $\rightarrow$ 7 .

8.3 Applying of LSM

8.3.1 Applying of SA Method

Example (5): Now we will use SA to solve example (1).

Let T=100; α=0.95; r=2,3,3,1,2,1,3,2,2,1,2;

Lets start with s=[3 2 1 4]; Cost=29;

1. r=2; s1 = Mutate (s)=[3 1 2 4]; NewCost = Evaluate (s1)=31;
   ΔCost = NewCost −Cost= 31 - 29 = - 2;
   p (Δ,T) = 0.9802; Rand=0.2;
   if (ΔCost ≤ 0) OR (p (Δ,T) > Rand)
   Cost = NewCost=31;
   s = s1; T=T* α = 95;
2. r=3; s1 = Mutate (s)=[3 1 4 2]; NewCost = Evaluate (s1)=27;
   ΔCost = NewCost −Cost= 27 - 31 = - 4;
   p (Δ,T)= $e$xp(-Δ$Cost$/T) = 1.043; Rand=0.2;
   if (ΔCost ≤ 0) OR (p (Δ,T) > Rand)
   Cost = NewCost=27;
   s = s1; T=T* α = 90.25;
3. r=3; s1 = Mutate (s)=[3 1 2 4]; NewCost = Evaluate (s1)=31;
   ΔCost = NewCost −Cost= 31 - 27 = 4;
   p (Δ,T) = 0.9566; Rand=0.2;
   if (ΔCost ≤ 0) OR (p (Δ,T) > Rand)
   Cost = NewCost=31;
   s = s1; T=T* α = 85.7375;
4. r=1; s1 = Mutate (s)=[1 3 2 4]; NewCost = Evaluate (s1)=31;
   ΔCost = NewCost −Cost= 31 - 31 = 0;
   p (Δ,T) = 1; Rand=0.2;
   if (ΔCost ≤ 0) OR (p (Δ,T) > Rand)
   Cost = NewCost=31;
   s = s1; T=T* α = 81.4506;
5. r=2; s1 = Mutate (s)=[1 2 3 4]; NewCost = Evaluate (s1)=24;
   ΔCost = NewCost −Cost= 24 - 31 = -7;
   p (Δ,T) = 1.0897; Rand=0.2;
   if (ΔCost ≤ 0) OR (p (Δ,T) > Rand)
   Cost = NewCost=24;
   s = s1; T=T* α = 77.3781;
6. r=1; s1 = Mutate (s)=[2 1 3 4]; NewCost = Evaluate (s1)=23;
   ΔCost = NewCost −Cost= 23 - 24 = -1;
   p (Δ,T) = 1.013; Rand=0.2;

if (ΔCost $\le$ 0) OR (p (Δ,T) > Rand)
Cost = NewCost=23;
s = s1; T=T* α = 73.5092;

and so on… after 100 iterations we obtain **s1=[4 1 2 3]**, with best solution **Cost=7**.

8.3.2 Applying of PSO Method

Example (6): Now we will use PSO using example (6).

n=4, MI=100 {*max. iteration*}, PS=4 {*pop. Size*}, $c_1 = c_2 = 2$,
$V \in [V_{min}, V_{max}] = [-4,4]$, {*max velocity*}. $X \in [a,b] = [-1,1]$; {*max position*}
$\omega = [0.4,0.9] = 0.4 + \frac{0.5(MI-i)}{MI}$, G=1;

Initialization:

$$X = \begin{bmatrix} 0.6 & 0.8 & -0.7 & 0.9 \\ -0.7 & 0.8 & 0.7 & 0.9 \\ 0.1 & -0.8 & 0.6 & 0.9 \\ 0.3 & 0.5 & 0.4 & -0.2 \end{bmatrix} \in [-1,1], V = \begin{bmatrix} 1.1 & -3.2 & -1.8 & 0.4 \\ 3.6 & 0.1 & 2.4 & -2.8 \\ 1.2 & -3.7 & 2.8 & 3.5 \\ 1.2 & -2.6 & 1.6 & -3.7 \end{bmatrix} \in [-4,4]$$

For i=1 : MI {=4}

1. i=1, $\omega = 0.4 + \frac{0.5(100-1)}{100} = 0.895.$

Fitness: For j=1:PS {=4}

$X_1 = [0.6,0.8,-0.7,0.9], p_1 = [3,1,2,4], F_1 = 31, Bp_1 = p_1, Xp_1 = X_1.$

$p_2 = [1,3,2,4],\ F_2 = 31, Bp_2 = p_2, Xp_2 = X_2.$

$p_3 = [2,1,3,4],\ F_3 = 23, Bp_3 = p_3, Xp_3 = X_3.$

$p_4 = [4,1,2,3],\ F_4 = 15, Bp_4 = p_4, Xp_4 = X_4.$

$\boldsymbol{Bestp} = [\mathbf{4,1,2,3}],\ \boldsymbol{BestF} = \mathbf{15}, \boldsymbol{G} = \mathbf{4}.$

Evolution: $r_1 = [0.2,0.4,0.6,0.8];\ r_2 = [0.3,0.5,0.7,0.9];$

$$V_j = w * V_j + c_1 * r_1 * (Xp_j - X_j) + c_2 * r_2 * (Xp_G - X_j) \qquad \ldots(1)$$

$$X_j = X_j + V_j \qquad \ldots(2)$$

For $\boldsymbol{j} = \mathbf{1}$: $\boldsymbol{PS}$ {= 4}

$j = 1;$

For $\boldsymbol{k} = \mathbf{1}$: $\boldsymbol{n}$ {= 4}

$k = 1$

$V_{11} = 0.895 * 1.1 + 2 * 0.2 * (0.6 - 0.6) + 2 * 0.3 * (0.6 - 0.3) = 1.1645$

If $V_{11} > V_{max}$ then $V_{11} = V_{11} - V_{max}$

If $V_{11} < V_{min}$ then $V_{11} = V_{11} + V_{max}$

$V_{11} = 1.1645$

$X_{11} = X_{11} + V_{11} = 0.6 + 1.1645 = 1.7645$

If $X_{11} > b$ then $X_{11} = X_{11} - b$

If $X_{11} < a$ then $X_{11} = X_{11} + b$

$X_{11} > 1$, then $X_{11} = 1.7645 - 1 = 0.7645$.
$k = 2;\ V_{12} = -3.164, X_{12} = -0.364$
$k = 3;\ V_{13} = -0.071, X_{13} = -0.771$
$k = 4;\ V_{14} = -1.622, X_{14} = -0.722$
$\boldsymbol{X_1 = [0.7645, -0.364, -0.771, -0.722]}$
$\boldsymbol{V_1 = [1.1645, -3.164, -0.071, -1.622]}$
$j = 2;$
$\boldsymbol{X_2 = [0.122, 0.5895, 0.482, 0.414]}$
$\boldsymbol{V_2 = [3.822, -3.2105, 1.782, -0.486]}$
$j = 3;$
$\boldsymbol{X_3 = [0.294, -0.8115, 0.826, 0.0525]}$
$\boldsymbol{V_3 = [1.194, -2.0115, 2.226, 1.1525]}$
$j = 4;$
$\boldsymbol{X_4 = [0.374, -0.827, 0.832, -0.5115]}$
$\boldsymbol{V_4 = [1.074, -2.327, 1.432, -3.3115]}$

2. i=2, $\omega = 0.4 + \frac{0.5(100-2)}{100} = 0.89$.

Fitness: For j=1:PS {=4}
$p_1 = [3,4,2,1], F_1 = 22, Xp_1 = X_1$.
$p_2 = [1,4,3,2],\ F_2 = 17, Xp_2 = X_2$.
$p_3 = [2,4,1,3],\ F_3 = 10, Xp_3 = X_3$.
$p_4 = [2,4,1,3],\ F_4 = 10, Xp_4 = X_4$
$\boldsymbol{Bestp = [2,4,1,3],\ BestF = 10, G = 4}$.
Evolution: and so on….

The best solution after 2 iterations is $\boldsymbol{Z = 10}$, for schedule $[\mathbf{2}, \mathbf{4}, \mathbf{1}, \mathbf{3}]$. and so on… after 100 iterations we obtain the schedule **[4 1 2 3]**, with best solution **Cost=Z=7**.