Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2025 (43)** : 1–17.
ISSN-0037-8712
doi:10.5269/bspm.76378

A Comparative Study of N*2 FSSP with Transportation Time & Processing Time as Trapezoidal Fuzzy Number

Pooja Kaushik, Sonia Goel*, Deepak Gupta

ABSTRACT: For the industrial and service industries to use resources efficiently, scheduling is crucial. Scheduling issues can vary widely depending on the industry and production setting. In these production circumstances, n jobs are processed by m separate machines. Finding a precise solution to scheduling issues becomes more challenging as the number of jobs and machines used increases. This study describes the bi stage flow shop scheduling problem when there are parallel equipotential processors at first stage and single at second stage with transportation time included and processing time as trapezoidal fuzzy number. The effectiveness of heuristics is compared in finding the minimum makespan. This study focuses on the comparative study of B&B, Particle Swarm Optimization (PSO) and Palmar method for solving the two stage FSSP under fuzzy processing times and transportation times. The main goal is to reduce the makespan while determining the optimal job sequence.

Key Words: heuristics, scheduling, trapezoidal fuzzy number (TrFNs), transportation technique, PSO, Palmar, Bi stage FSSP.

Contents

* Corresponding author.

Submitted April 15, 2025. Published September 17, 2025
2020 *Mathematics Subject Classification:* 90B35,68M20.

1. Introduction

Applications of fuzzy optimization techniques in management and engineering has garnered a lot of attention lately due to their high accuracy, efficiency, and adaptability, which yield realistic results of high quality. This have been extensively studied in the fields of industry, engineering, and health. Flow shop scheduling is a fundamental problem in production and operations management, where a set of jobs must be processed sequentially on multiple machines. The two-stage FSSP is a simplified version where jobs pass through the machines in the same order. In real-world manufacturing and logistics scenarios, uncertainties in processing times and transportation times between stages are common, making deterministic models less effective. To address these uncertainties, fuzzy numbers particularly TrFNs, have been widely used. By incorporating trapezoidal fuzzy numbers to model uncertainties, this study provides a more realistic approach to scheduling in uncertain environments. The comparative analysis will help to determine the effectiveness of these algorithms in terms of solution quality, computational efficiency and robustness under uncertainty. The two-stage FSSP has been extensively studied due to its practical applications in manufacturing, logistics, and service industries. Early research on flow shop scheduling began with Johnson's (1954) [7] rule, which provided an optimal solution for minimizing the makespan in a two-machine deterministic environment. However, real-world scheduling problems involve additional complexities, such as transportation time between machines and uncertain processing times, which necessitate more advanced optimization approaches. To address these challenges, exact and heuristic methods have been explored in the literature. Gupta & Stafford (2006) [3] used B&B, which is a widely used exact algorithm for solving scheduling problems. It systematically explores the solution space by eliminating suboptimal branches, ensuring an optimal solution. However, its computational complexity increases exponentially with problem size, making it inefficient for large-scale scheduling scenarios. With the advancement of computational intelligence, metaheuristic approaches have gained popularity in solving complex scheduling problems. PSO, introduced by Kennedy and Eberhart (1995) [8], is a population-based optimization technique inspired by the collective behaviour of birds and fish. PSO has been successfully applied to scheduling problems, demonstrating faster convergence and robustness compared to traditional methods by Kuo et al. (2009) [10]. Unlike B&B, which guarantees an optimal solution but is computationally expensive, PSO efficiently explores the solution space to provide near-optimal solutions. Several studies have shown that PSO outperforms conventional algorithms in solving scheduling problems under uncertainty, making it a viable alternative for large-scale flow shop scheduling. Palmer (1965) [11] employed a single iteration approach using the slop index to determine the minimum sequence and achieve the necessary outcome of decreasing the make span. Uncertainty in processing times is a critical challenge in scheduling problems, as real-world environments often involve variability due to machine breakdowns, operator efficiency or external factors. Fuzzy set theory, introduced by Zadeh (1965) [14], has been extensively used to model imprecise data in scheduling. Among different fuzzy representations, trapezoidal fuzzy numbers (TFNs) have been widely adopted due to their computational efficiency and ability to model uncertainty in processing times by Dubois & Prade (1979) [2]. Research by Wang et al. (2016) [12] and Goyal et al. (2021) [9] has demonstrated that incorporating fuzzy processing times leads to more realistic scheduling models that better reflect actual production conditions. Furthermore, Hundal et al. (1988) [13] worked on Palmer's heuristic. Studies by Chen et al. (2007) [1] have explored the impact of transportation time in fuzzy scheduling, highlighting the need for optimization techniques that can effectively handle such uncertainties. Two and three-stage FSSP with equipotential machines using branch and bound method was studied by Sonia Goel et al. (2018) (2022) [4], [5], [6]. Pooja Kaushik et al. (2025) [15] also worked on comparative study of B&B with heuristics NEH and CDS. Despite the extensive research on flow shop scheduling, a direct comparative analysis of B&B and PSO in solving two stage FSSP with fuzzy processing times and transportation times remains limited. While B&B ensures optimality, its practical applicability is constrained by computational demands. In contrast, PSO offers a scalable and efficient alternative for solving complex scheduling problems under uncertainty. This study aims to fill this gap by evaluating the performance of B&B and PSO in terms of makespan minimization, computational efficiency, and robustness in handling fuzzy processing times and transportation times. The findings of this research will provide valuable insights into selecting appropriate optimization techniques for real-world scheduling applications.

2. Preliminaries

Definition 2.1 *Fuzzy Subset Let $\grave{E}$ be an finite or infinite set. Let $\mathring{A} \subseteq \grave{E}$. Then the set of ordered pairs $\left(x, \mu_{\mathring{A}}(x)\right)$ gives the fuzzy subset $\mathring{A}$ of $\grave{E}$, where $x \in E$ and $\mu_{\mathring{A}}(x)$ is the degree of membership of x in $\grave{E}$.*

Definition 2.2 *Fuzzy Number A map $f_a : R \to \left[0, 1\right]$ such that $f_a(x) = 1, \Leftrightarrow x = a$ and $Lim_{|x|\to} f_a(x) = 0$ is fuzzy number.*

Definition 2.3 *A fuzzy number $\mathring{A} = \left(\grave{a}, ß, \acute{c}\right)$ is called triangular fuzzy number (graphical form is given in figure 1) if its membership function satisfy the following conditions*

1. *$\grave{a}$ to ß is an increasing function*

2. *ß to $\acute{c}$ is decreasing function*

3. *$\grave{a} \leq ß \leq \acute{c}$*

and is given by

$$\mu_{\mathring{A}}(x) = \begin{cases} 0 & for\ x < \grave{a} \\ \frac{x-\grave{a}}{ß-\grave{a}} & for\ \grave{a} \leq x \leq ß \\ \frac{\acute{c}-x}{\acute{c}-ß} & for\ ß \leq x \leq \acute{c} \\ 0 & for\ x > \acute{c} \end{cases}$$

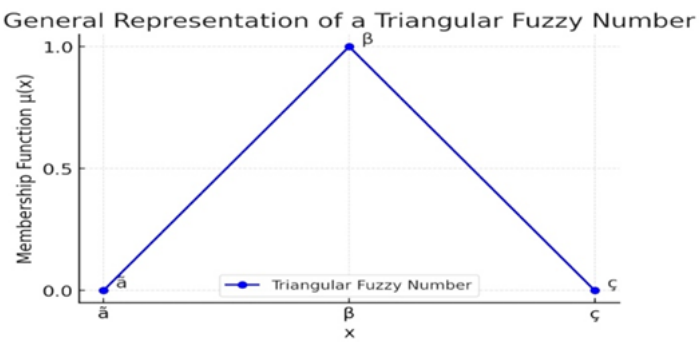

Figure 1: Graph of triangular fuzzy number

Definition 2.4 *Trapezoidal Fuzzy Number $\mathring{A} = \left(\grave{a}, ß, \acute{c}, d\right)$ is a fuzzy number if its membership function is defined by and its graphical form is given in figure 2.*

$$\mu_{\mathring{A}}(x) = \begin{cases} \frac{x-\grave{a}}{ß-\grave{a}} & for\ \grave{a} \leq x \leq ß \\ 1 & for\ ß \leq x \leq \acute{c} \\ \frac{d-x}{d-\acute{c}} & for\ \acute{c} \leq x \leq d \\ 0 & otherwise \end{cases}$$

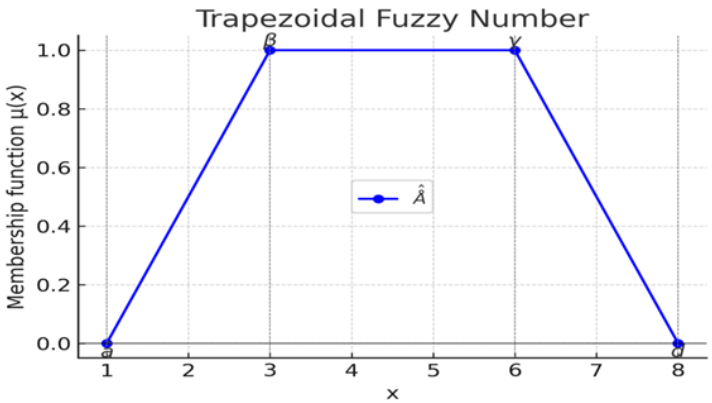

Figure 2: Graph of trapezoidal fuzzy number

3. Practical Situation

In the food processing industry, scheduling is commonly used to optimize production efficiency. In the first stage, multiple identical machines operate in parallel to handle initial processing, such as mixing, grinding, or baking. For example, in a biscuit manufacturing unit, multiple ovens bake the dough simultaneously to increase throughput. However, after this stage, all processed items move to the second stage, where a single specialized machine performs the final operation i.e. packaging. Since only one machine is available at this stage, it often creates a bottleneck, leading to waiting times and production delays. To handle the uncertainty in processing times caused by factors like ingredient variations, machine performance, or environmental conditions, trapezoidal fuzzy numbers are used to represent the processing durations more realistically. Efficient scheduling of jobs across the parallel machines at the first stage and proper sequencing at the second stage are crucial to minimizing idle time and maximizing productivity. This type of scheduling model is widely applied in industries such as dairy processing, beverage manufacturing, confectionery production, and packaged food manufacturing, where balancing production speed, quality, and efficiency is essential for profitability and customer satisfaction.

In the food processing industry, first stage consists of multiple parallel machines performing initial processing, and the second stage has a single machine responsible for final operations like packaging or quality control. Below are the detailed steps involved at both stages with an example of biscuit manufacturing.

Stage 1: Initial Processing (Parallel Machines)
At this stage, raw ingredients are processed using multiple identical machines to increase efficiency.
Steps:

1. Ingredient Mixing: Raw materials (flour, sugar, butter, etc.) are mixed in large industrial mixers.
2. Dough Rolling & Cutting: The dough is rolled and cut into biscuit shapes using automated cutters.
3. Baking (Parallel Machines): Multiple ovens operate simultaneously to bake the biscuits.
4. Transfer to Cooling Conveyor: Once baked, biscuits move to a cooling conveyor before final processing.

Since multiple machines work in parallel, jobs must be assigned optimally to balance the load and minimize idle time. However, processing time may vary due to ingredient consistency, oven temperature fluctuations, or dough moisture levels, which can be represented using trapezoidal fuzzy numbers.

Stage 2: Final Processing (Single Machine)
After initial processing, the products go through a single specialized machine for the final stage. Biscuits

are cooled to prevent breakage and moisture retention. A single inspection machine ensures uniform size, shape, and texture. An automated system packs biscuit into containers or packets. The machine seals packets and applies labels with product details. The final product is moved to storage or sent for delivery.
Since only one machine handles final processing, this stage can become a bottleneck if jobs are not properly sequenced. Efficient scheduling ensures smooth workflow and minimizes waiting times.
This two-stage scheduling model is widely applicable in food industries like dairy processing, beverage production, packaged snacks, and bakery items, ensuring high-speed production with minimal delays.

4. Mathematical model

Suppose there are n tasks to be performed on machines $\ddot{A}$ and $\ddot{B}$. First stage consists of machine $\ddot{A}$ having three parallel equipotential machines and single machine at second stage given in table 1. Let $(\varsigma_i, d_i, e_i, f_i)$ be the fuzzy utilisation time of i^{th} task on $\ddot{A}$ and (g_i, h_i, k_i, l_i) be the fuzzy utilisation time of i^{th} task on $\ddot{B}$. Each task can be done on any of the three parallel equipotential machines not necessarily on all of them and then for further processing the task is transformed to single machine at second stage. The goal of the proposed work is to schedule the sequencing of jobs at two stages under more practical environment with utilisation time as trapezoidal fuzzy number to optimize the elapse time and cost of processing of jobs in a more efficient manner.

Table 1: Tabular form of model

Job	Processor $\ddot{A}$		Processor $\ddot{B}$	
	$\ddot{A}_1$ $\ddot{A}_2$ $\ddot{A}_3$ $\ddot{A}_n$	Processing time of $\ddot{A}$ (k_i)	Transportation time T_i	Processing time of $\ddot{B}$ (p_i)
1	$\grave{a}_{11}$ $\grave{a}_{12}$ $\grave{a}_{13} \cdots \grave{a}_{1n}$	$(\varsigma_1, d_1, e_1, f_1)$	T_1	(g_1, h_1, k_1, l_1)
2	$\grave{a}_{21}$ $\grave{a}_{22}$ $\grave{a}_{23} \cdots \grave{a}_{2n}$	$(\varsigma_2, d_2, e_2, f_2)$	T_2	(g_2, h_2, k_2, l_2)
3	$\grave{a}_{31}$ $\grave{a}_{32}$ $\grave{a}_{33} \cdots \grave{a}_{3n}$	$(\varsigma_3, d_3, e_3, f_3)$	T_3	(g_3, h_3, k_3, l_3)
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	$\grave{a}_{n1}$ $\grave{a}_{n2}$ $\grave{a}_{n3}$ $\grave{a}_{nn}$	$(\varsigma_n, d_n, e_n, f_n)$	T_n	(g_n, h_n, k_n, l_n)
t_{pj}	t_{11} t_{12} t_{13} t_{1n}			

4.1. Notations

i : Index of the job
$(\varsigma_n, d_n, e_n, f_n)$: Fuzzy utilisation time of i^{th} task on K_j
(g_n, h_n, k_n, l_n) : Fuzzy utilisation time of i^{th} task on Machine E
$\ddot{A}_n$: Parallel equipotential machines for Machine K
t_{pj} : Total available time of parallel equipotential machines of type k for p=1,2,3
T_n : Transportation time of moving jobs from machine $\ddot{A}$ to $\ddot{B}$.

4.2. Assumptions

(a) There are n different, independent operation jobs that can be processed at time zero.

(b) Job descriptions are predetermined.

(c) All of the machines are always accessible.

(d) An operation runs uninterrupted once it starts.

(e) The tasks that need to be completed don't depend on one another.

(f) Pre-emption of employment are not allowed.

(g) No job interruptions are permitted, and once a task is started, it must be completed before another task can begin.

5. Algorithm

Step 1: Use Yager's ranking rule to find the crisp value

$$\bar{k}_i = \frac{1}{2}\Big[\big(ß_i + ç_i\big) - \frac{4}{5}\big(ß_i - å_i\big) + \frac{2}{3}\big(ỳ_i - ç_i\big)\Big]$$

where $\big(å_i, ß_i, ç_i, ỳ_i\big)$ is the trapezoidal fuzzy processing time
and

$$\bar{p}_i = \frac{1}{2}\Big[\big(è_i + f_i\big) - \frac{4}{5}\big(è_i - d_i\big) + \frac{2}{3}\big(J_i - f_i\big)\Big]$$

where $\big(d_i, è_i, f_i, J_i\big)$ is the trapezoidal fuzzy processing time.

Step 2. Apply transportation techniques VAM and MODI. To apply any transportation technique, we examine the constraint

$$\sum_{j=1}^{3} t_{1j} = \sum_{i=1}^{n} m_i$$

where $m_i = \bar{k}_i + T_i$.
If the problem is unbalanced, first balance it by adding dummy job or machines.

Step 3. Use branch and bound method by applying the rule

$$g' = max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in J'_k}\big(a_i\big) \qquad \& \qquad g'' = \max_{i \in J'_k} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$$

where $a_i = \bar{p}_i + T_i$ and J'_k denote the tasks not taken into consideration under branching.

Then evaluate $G = max(g', g'')$ for all the jobs.
Find the minimum of all values of G and start from that vertex as the first job in the optimal subsequence. Repeat the above process till we reach termination point of branch which will give us the most ideal optimal sequence of jobs.

6. Numerical problem

Table 2: Mathematical problem

Jobs	Professor $\ddot{A}$					Professor $\ddot{B}$	
	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	Processing time of $\ddot{A}$	Transportation on time T_i	Processing time of $\ddot{B}$
1	8	6	10	3	(2,7,8,11)	6	(3,5,7,8)
2	9	6	2	8	(4,8,12,16)	5	(4,6,7,9)
3	5	9	12	8	(3,8,9,12)	8	(5,8,10,11)
4	4	7	7	6	(5,10,12,15)	7	(4,5,7,9)
5	10	2	8	9	(6,9,11,14)	5	(6,8,11,13)
Available time	13	10	12	9.5			

Solution: Solution of the problem given in table 2 is as follows

Step 1: We apply the Yager's ranking formula to get the crisp value of processing time

Table 3: Numerical problem following the application of average high-ranking formula

Jobs	Professor $\ddot{A}$					Professor $\ddot{B}$	
	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	Processing time of $\ddot{A}$	Transportation on time T_i	Processing time of $\ddot{B}$
1	8	6	10	3	6.5	6	5.5
2	9	6	2	8	9.7	5	6.4
3	5	9	12	8	7.5	8	8.1
4	4	7	7	6	10	7	6.3
5	10	2	8	9	9.8	5	9.4
Available time	13	10	12	9.5			

Changing fuzzy trapezoidal number utilisation time into crisp value using

$$k_i = \frac{1}{2}\Big[\big(ß_i + ç_i\big) - \frac{4}{5}\big(ß_i - å_i\big) + \frac{2}{3}\big(ŷ_i - ç_i\big)\Big]$$

(2,7,8,11) is transformed into 6.5
(4,8,12,16) is transformed into 9.7
(3,8,9,12) is transformed into 7.5
(5,10,12,15) is transformed into 10
(6,9,11,14) is transformed into 9.8

Likewise (3,5,7,8) is transformed into 5
(4,6,7,9) is transformed into 5
(5,8,10,11) is transformed into 4
(4,5,7,9) is transformed into 6
(6,8,11,13) is transformed into 7

Complete numerical problem with crisp values is given in table 3.

Now we apply VAM and Modified Distribution Method (MODI) to further optimize the processing times of jobs and the reduced problem is given in table 4 below.

Table 4: Processing time of parallel equipotential machines after MODI

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_i + T_i$
1	0	0	0	6.5	11.5
2	0	0	9.7	0	11.4
3	7.5	0	0	0	16.1
4	4.5	2.2	0.3	3	13.3
5	0	9.8	0	0	14.4

Now we apply B&B algorithm and calculate the lower bound of jobs to obtain the sequence of jobs for optimal solution given in table 5.

Table 5: Lower bound of jobs

(i)	$g' = max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in J'_k}(a_i)$	$g'' = \max_{i \in J'_k} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$	$G = max\{g', g''\}$
1	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in 2,3,4,5}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in 2,3,4,5} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 6.5 + 66.7 = 73.2$	73.2
2	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in 1,3,4,5}(a_i)$ $= 12 + 11.5 = 23.5$	$\max_{i \in 1,3,4,5} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 9.7 + 66.7 = 76.4$	76.4
3	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in 1,2,4,5}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in 1,2,4,5} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 7.5 + 66.7 = 74.2$	74.2
4	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in 1,2,3,5}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in 1,2,3,5} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 4.5 + 66.7 = 71.2$	71.2
5	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in 1,2,3,4}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in 1,2,3,4} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 9.8 + 66.7 = 76.5$	76.5

Here, lower bound is 71.2 corresponding to job 4. So, fixing job 4 at 1^st place and continue the process till all jobs gets place in the optimal order as shown in table 6,7&8.

Table 6: Lower bound of jobs when job 4 is fixed at first place

(i)	$g' = max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in J'_k}(a_i)$	$g'' = \max_{i \in J'_k} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$	$G = max\{g', g''\}$
{4,1}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in \{2,3,5\}}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in \{2,3,4,5\}} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 9.5 + 66.7 = 76.2$	76.2
{4,2}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in \{1,3,5\}}(a_i)$ $= 12 + 11.5 = 23.5$	$\max_{i \in \{1,3,4,5\}} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 10 + 66.7 = 76.7$	76.7
{4,3}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in \{1,2,5\}}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in \{1,2,4,5\}} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 12 + 66.7 = 78.7$	78.7
{4,5}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i \in \{1,2,3,4\}}(a_i)$ $= 12 + 11.4 = 23.4$	$\max_{i \in \{1,2,3,4\}} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 12 + 66.7 = 78.7$	78.7

Table 7: Lower bound of jobs when 4^{th} & 1^{st} job is fixed

(i)	$g' = max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i\in J'_k}(a_i)$	$g'' = \max_{i\in J'_k} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$	$G = max\{g', g''\}$
{4,1,2}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min(a_i)$ $= 12 + 11.4 = 23.4$	$\max \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 9.5 + 66.7 = 76.2$	76.2
{4,1,3}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min(a_i)$ $= 12 + 11.5 = 23.5$	$\max \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 10 + 66.7 = 76.7$	76.7
{4,1,5}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min(a_i)$ $= 12 + 11.4 = 23.4$	$\max \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 12 + 66.7 = 78.7$	78.7

Table 8: Lower bound after fixing 4^{th}, 1^{st} and 2^{nd} job

(i)	$g' = max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min_{i\in J'_k}(a_i)$	$g'' = \max_{i\in J'_k} \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$	$G = max\{g', g''\}$
{4,1,2,3}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min(a_i)$ $= 12 + 14.4 = 26.4$	$\max \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 12 + 66.7 = 78.7$	78.7
{4,1,2,5}	$max\left(\sum_{i=1}^{n} \ddot{A}_{ij}\right) + \min(a_i)$ $= 12 + 16.1 = 28.1$	$\max \ddot{A}_{ij} + \sum_{i=1}^{n} a_i$ $= 12 + 66.7 = 78.7$	78.7

Therefore {4,1,2,3,5} is the required optimal sequence. Table 9 gives best feasible solution.
Note: {4,1,2,5,3} is also an optimal sequence.

Table 9: In-out table

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
4	$0 - 4.5$	$0 - 2.2$	$0 - 0.3$	$0 - 3$	$4.5 - 17.8$
1	–	–	–	$3 - 9.5$	$17.8 - 29.3$
2	–	–	$0.3 - 10$	–	$29.3 - 40.7$
3	$4.5 - 12$	–	–	–	$40.7 - 56.8$
5	–	$2.2 - 12$	–	–	$56.8 - 71.2$

Therefore, elapse time is 71.2 units.

7. Gantt Chart

Solution obtained in table 9 is represented by Gantt chart given in figure 3.

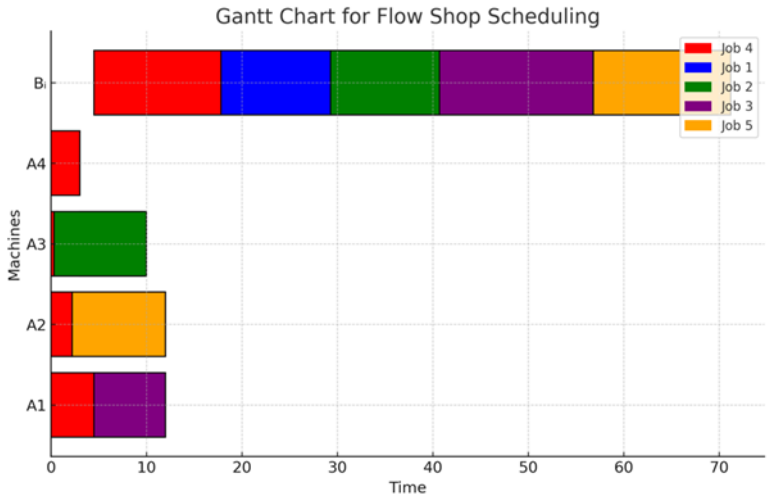

Figure 3: Gantt chart

8. Solution by PSO

The social behaviour of fish schools and flocks of birds served as the inspiration for the well-known optimization algorithm known as particle swarm optimization (PSO). It is a popular population-based search technique for resolving challenging optimization issues. Russell Eberhart and James Kennedy initially presented it in 1995.

8.1. Basic Concepts

1. Particles: A particle is a potential solution to the optimization problem. Each particle has a position, velocity, and fitness value.

2. Swarm: A swarm is a collection of particles. The swarm is used to search for the optimal solution.

3. Fitness Function: The fitness function is used to evaluate the quality of each particle's position. The fitness function is problem-dependent.

4. Personal Best (pbest): The personal best position of a particle is the best position it has visited so far.

5. Global Best (gbest): The global best position is the best position visited by any particle in the swarm.

8.2. Parameters

- Inertia Weight (w): The inertia weight controls the trade-off between exploration and exploitation. A high value of w encourages exploration, while a low value encourages exploitation.

- Cognitive Learning Rate (c1): The cognitive learning rate controls the step size of the particle's movement towards its personal best position.

- Social Learning Rate (c2): The social learning rate controls the step size of the particle's movement towards the global best position.

- Population Size: It controls the number of particles in the swarm.

- Maximum Number of Iterations: It controls the stopping criterion for the algorithm.

8.3. Solution

To solve this FSSP using PSO and find the minimum makespan, we'll follow these steps:

Step 1: Initialisation
We have 5 jobs that need to be processed on two processors $\ddot{A}$ and $\ddot{B}$. Processor $\ddot{A}$ consists of four processors i.e. $\ddot{A}_1, \ddot{A}_2, \ddot{A}_3, \ddot{A}_4, \ddot{A}_5$ for each job. Processor $\ddot{B}$ starts working on a job only after it has completed processing on A and has undergone a transportation delay. The objective is to determine the optimal job sequence that minimizes the total completion time.
Since we have 5 jobs, we need to initialize N particles (random sequences of jobs). Let's assume 10 particles. Each particle represents a possible sequence of jobs. For example:

$$P_1 \to [J_1, J_2, J_3, J_4, J_5]; P_2 \to [J_3, J_1, J_2, J_4, J_5]; P_3 \to [J_2, J_3, J_1, J_5, J_4]; P_4 \to [J_5, J_3, J_2, J_1, J_4];$$

$P_5 \to [J_4, J_5, J_1, J_3, J_2]$;
Each particle position is a permutation vector, and velocities update the sequence. Each particle has an associated velocity, which determines how jobs are swapped.

Step 2: Evaluate Fitness (Calculate Makespan)
We calculate the makespan for each particle using the flow shop scheduling method given in table 10,11,12,13 & 14.

Table 10: Makespan for P_1

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
3	$0-7.5$	–	–	–	$7.5-23.6$
2	–	–	$0-9.7$	–	$23.6-35$
1	–	–	–	$0-6.5$	$35-46.5$
4	$7.5-12$	$0-2.2$	$9.7-10$	$6.5-9.5$	$46.5-59.8$
5	–	$2.2-12$	–	–	$59.8-74.2$

Table 11: Makespan for P_2

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
2	–	–	$0-9.7$	–	$9.7-21.1$
3	$0-7.5$	–	–	–	$21.1-37.2$
1	–	–	–	$0-6.5$	$37.2-53.3$
5	–	$0-9.8$	–	–	$53.3-67.7$
4	$7.5-12$	$9.8-12$	$9.7-10$	$6.5-9.5$	$67.7-81.0$

Table 12: Makespan for P_3

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
5	–	$0-9.8$	–	–	$9.8-24.2$
3	$0-7.5$	–	–	–	$24.2-40.3$
2	–	–	$0-9.7$	–	$40.3-51.4$
1	–	–	–	$0-6.5$	$51.4-62.9$
4	$7.5-12$	$9.8-12$	$9.7-10$	$6.5-9.5$	$62.9-76.2$

Table 13: Makespan for P_4

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
4	$0-4.5$	$0-2.2$	$0-0.3$	$0-3$	$4.5-17.8$
5	–	$2.2-12$	–	–	$17.8-32.2$
1	–	–	–	$3-9.5$	$32.2-43.7$
3	$4.5-12$	–	–	–	$43.7-59.8$
2	–	–	$0.3-10$	–	$59.8-71.2$

Table 14: Makespan for P_5

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
2	–	–	$0-9.7$	–	$9.7-21.1$
1	–	–	–	$0-6.5$	$21.1-32.6$
3	$0-7.5$	–	–	–	$32.6-48.7$
4	$7.5-12$	$0-2.2$	$9.7-10$	$6.5-9.5$	$48.7-62$
5	–	$2.2-12$	–	–	$62-76.4$

Step 3. Update Personal Best (pbest)
Each particle records its pbest sequence which is given in table 15.

Table 15: Personal best of each particle

Practicals	pbest positions
P_1	74.2
P_2	81
P_3	76.2
P_4	71.2
P_5	76.4

Step 4. Update Global Best
The best sequence so far is $P_4(J_4, J_5, J_1, J_3, J_2)$ with makespan 71.2. Gbest is P_4's sequence.

Step 5. Update Velocity & Swap Jobs
Let us swap the particles and find makespan closure to gbest. After swapping jobs, we recalculate the makespan for each updated sequence.
Let's update Particle P_1 i.e. $\{J_3, J_2, J_1, J_4, J_5\}$. Suppose the velocity suggests swapping J_1 and J_3, the new sequence becomes: $\{J_1, J_2, J_3, J_4, J_5\}$ shown in table 16.

Table 16: Makespan after swapping job 1 and 3 in P_1

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
1	–	–	–	0 – 6.5	6.5 – 18
2	–	–	0 – 9.7	–	18 – 29.4
3	0 – 7.5	–	–	–	29.4 – 45.5
4	7.5 – 12	0 – 2.2	9.7 – 10	6.5 – 9.5	45.5 – 58.5
5	–	2.2 – 12	–	–	58.5 – 73.2

P_2 improves from 81 → 76.5, so it updates pbest given in table 17.

Table 17: Makespan after swapping job 2 and 5 in P_2

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
5	–	0 – 9.8	–	–	9.8 – 24.2
3	0 – 7.5	–	–	–	24.2 – 40.3
1	–	–	–	0 – 6.5	40.3 – 51.8
2	–	–	0 – 9.7	–	51.8 – 63.2
4	7.5 – 12	9.8 – 12	9.7 – 10	6.5 – 9.5	63.2 – 76.5

Similarly, particle P_3 updates by moving closer to gbest given in 18.

Table 18: Makespan after swapping job 5 and 1 in P_3

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
1	–	–	–	0 – 6.5	6.5 – 18
3	0 – 7.5	–	–	–	18 – 34.1
2	–	–	0 – 9.7	–	34.1 – 45.5
5	–	0 – 9.8	–	–	45.5 – 59.9
4	7.5 – 12	9.8 – 12	9.7 – 10	6.5 – 9.5	59.9 – 73.2

On the same lines, Particle $P_4\{J_4, J_5, J_1, J_3, J_2\}$ and $P_5\{J_3, J_1, J_2, J_4, J_5\}$ updates by moving closer to gbest given in 19 and 20.

Table 19: Makespan after swapping job 5 and 2 in P_4

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
4	0 – 4.5	0 – 2.2	0 – 0.3	0 – 3	4.5 – 17.8
2	–	–	0.3 – 10	–	17.8 – 29.2
1	–	–	–	3 – 9.5	29.2 – 40.7
3	4.5 – 12	–	–	–	40.7 – 56.8
5	–	2.2 – 12	–	–	56.8 – 71.2

Table 20: Makespan after swapping job 2 and 3 in P_5

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
3	$0-7.5$	–	–	–	$7.5-23.6$
1	–	–	–	$0-6.5$	$23.6-35.1$
2	–	–	$0-9.7$	–	$35.1-46.5$
4	$7.5-12$	$0-2.2$	$9.7-10$	$6.5-7.5$	$46.5-59.8$
5	–	$2.2-12$	–	–	$59.8-74.2$

This updated makespan after swapping is given in table 21.

Table 21: pbest of each particle after swapping

Practicals	pbest positions
P_1	73.2
P_2	76.5
P_3	73.2
P_4	71.2
P_5	74.4

Step 6: Repeat Until Convergence
This process continues for many iterations until no further improvement happens. After many iterations, the best sequence comes out to be $\{J_4, J_5, J_1, J_3, J_2\}$.

9. Palmer Method

Palmer's heuristic is a priority rule-based scheduling method that assigns a priority index (slope index) to each job based on processing times across machines. The fundamental assumption is that jobs with a decreasing trend in processing times across machines should be scheduled earlier to reduce bottlenecks. Slope index (SI) is computed for all jobs, the jobs are sorted in descending order of SI and scheduled in that sequence. This prioritization favours jobs that have a decreasing processing time trend over machines, which helps balance workload and reduces machine idleness.

Solution: Consider tabular representation of problem in reduced form following time allocation to parallel equipotential machines is given in table 22.

Table 22: Time allocation on equipotential machines

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_i + T_i$
1	0	0	0	6.5	11.5
2	0	0	9.7	0	11.4
3	7.5	0	0	0	16.1
4	4.5	2.2	0.3	3	13.3
5	0	9.8	0	0	14.4

Reduced problem after taking average of processing times of parallel equipotential machines at first stage is given in table 23. Then, we calculate weights using the formula

$$w_i = (m - 2i + 1)$$

Where m is the number of machines and P_{ij} is the processing time of job (j) on machine (i).
Taking weights $w_1 = -1$ and w_2, Compute weighted sum of time for each job using the formula $p_w = \sum_j w_j p_{ij}$ given in table 24.

Table 23: Reduced problem

Jobs	$\ddot{A}_i$	$\ddot{B}_l + T_i$
1	1.625	11.5
2	2.425	11.4
3	1.875	16.1
4	2.5	13.3
5	2.45	14.4

Table 24: Weighted sum of jobs

Jobs	p_w	Weighted sum
1	p_{w1}	9.875
2	p_{w2}	8.975
3	p_{w3}	14.225
4	p_{w4}	10.8
5	p_{w5}	11.95

Sort Jobs: Arranging the jobs in descending order of SI's we get order $J_3 \to J_5 \to J_4 \to J_1 \to J_2$. Now, we calculating minimum makespan using in-out table 25.

Table 25: in-out table

Jobs	$\ddot{A}_1$	$\ddot{A}_2$	$\ddot{A}_3$	$\ddot{A}_4$	$\ddot{B}_l$
3	$0 - 7.5$	–	–	–	$7.5 - 23.6$
5	–	$0 - 9.8$	–	–	$23.6 - 38$
4	$7.5 - 12$	$9.8 - 12$	$0 - 0.3$	$0 - 3$	$38 - 51.3$
1	–	–	–	$3 - 9.5$	$51.3 - 62.8$
2	–	–	$0.3 - 10$	–	$62.8 - 74.2$

Hence, the makespan is 74.2 units.

9.1. Comparison between B&B, PSO and Palmar:

Comparison between the three heuristics is given in table 26 and figure 4.

Table 26: Comparison table

Iteration	**B&B**	**PSO**	**Palmar**
1	71.2	73.2	74.2
2	71.2	76.5	74.2
3	71.2	73.2	74.2
4	71.2	71.2	74.2
5	71.2	74.2	74.2

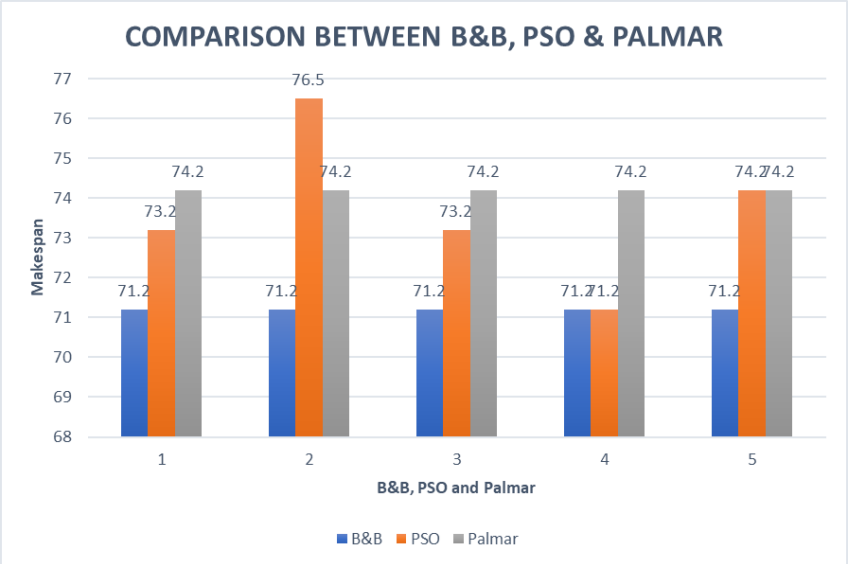

Figure 4: Comparison graph

10. Conclusion

In a nutshell, B&B, PSO and Palmar offer distinct advantages and trade-offs. B&B is a deterministic and exact method that guarantees an optimal solution, making it suitable for small-scale problems. However, its computational complexity grows exponentially, making it impractical for large-scale FSSP. On the other hand, PSO is a metaheuristic approach that provides near-optimal solutions efficiently, making it more scalable and adaptable to real-world scheduling problems. While PSO does not guarantee optimality, it significantly reduces computation time and can handle large problem instances with ease. Overall, B&B is ideal when absolute optimality is required for small problems, while PSO is the preferred choice for large-scale scheduling due to its speed, scalability, and adaptability. The Palmar Method, on the other hand, follows predefined rules to construct feasible solutions quickly but does not always ensure optimality. Compared to B&B and PSO, the Palmar Method is computationally lightweight and designed for practical efficiency rather than exhaustive exploration. While B&B is best suited for combinatorial optimization problems and integer programming, PSO is widely used in continuous optimization, machine learning, and engineering applications, whereas the Palmar Method is more specialized for scheduling and operational research.

11. Future scope

Future research can focus on developing hybrid B&B-PSO models that combine the accuracy of B&B with the scalability of PSO for improved scheduling performance. Additionally, integrating these algorithms with machine learning and cloud computing can enable real-time scheduling based on IoT data. Moreover, multi-objective optimization can optimize not only makespan but also energy consumption, cost, and machine efficiency. The integration of these scheduling algorithms with technologies, blockchain, and smart automation can make production systems more autonomous and secure. Overall, future research will focus on enhancing the speed, accuracy, and adaptability of these algorithms, leading to the development of more efficient and cost-effective scheduling systems in industries.

References

1. Chen, S.-J. and Hsu, M. C., *Fuzzy risk analysis based on the ranking of generalized trapezoidal fuzzy numbers*, Appl. Intell. **26**, 1–11 (2007).

2. Dubois, D. and Prade, H., *Fuzzy real algebra: some results*, Fuzzy Sets Syst. **2**, 327–348 (1979).

3. Gupta, J. N. D. and Stafford, E. F., *Flowshop scheduling research after five decades*, Eur. J. Oper. Res. **169**, 699–711 (2006).

4. Gupta, D. and Goel, S., *Three stage flow shop scheduling model with m-equipotential machines*, Int. J. Future Revolut. Comput. Sci. Commun. Eng. **4**, 269–274 (2018).

5. Gupta, D., Goel, S. and Mangla, N., *Optimization of production scheduling in two stage flow shop scheduling problem with m equipotential machines at first stage*, Int. J. Syst. Assur. Eng. Manag. **13**, 1162–1169 (2022).

6. Gupta, D. and Goel, S., *Branch and bound technique for two stage flow shop scheduling model with equipotential machines at every stage*, Int. J. Oper. Res. **44**, 462–472 (2022).

7. Johnson, S. M., *Optimal two- and three-stage production schedules with setup times included*, Naval Res. Logist. Q. **1**, 61–68 (1954).

8. Kennedy, J. and Eberhart, R., *Particle swarm optimization*, Proc. Int. Conf. Neural Netw. (ICNN'95), **4**, 1942–1948 (1995).

9. Goyal, B. and Kaur, S., *Specially structured flow shop scheduling models with processing times as trapezoidal fuzzy numbers to optimize waiting time of jobs*, Soft Comput. Probl. Solving **2**, 27–42 (2021).

10. Kuo, I.-H., Hoong, S.-J., Kao, T.-W., Lin, T.-L., Lee, C.-L., Terano, T. and Pan, Y., *An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model*, Expert Syst. Appl. **36**, 7027–7032 (2009).

11. Palmer, D. S., *Sequencing jobs through a multi-stage process in the minimum total time: a quick method of obtaining a near optimum*, J. Oper. Res. Soc. **16**, 101–107 (1965).

12. Wang, Y., Zhao, X. and Liu, F., *A fuzzy approach to two-stage flow shop scheduling with uncertain processing times*, Appl. Soft Comput. **47**, 304–315 (2016).

13. Hundal, T. S. and Rajgopal, J., *An extension of Palmer's heuristic for the flow shop scheduling problem*, Int. J. Prod. Res. **26**, 1119–1124 (1988).

14. Zadeh, L. A., *Fuzzy sets*, Inform. Control **8**, 338–353 (1965).

15. Kaushik, P., Gupta, D. and Goel, S., *Comparative study of B&B with heuristics NEH and CDS for bi-stage flow shop scheduling problem under fuzzy environment*, J. Emerg. Technol. Innov. Res. Math. Sci. **20**, 73–89 (2025).

Pooja Kaushik, ,
Department of Mathematics,
Maharishi Markandeshwar Engineering College,
Maharishi Markandeshwar (Deemed to be University),
Mullana (Ambala), Haryana, India
E-mail address: drtobekaushik2024@gmail.com, ORCID: 0009-0000-2799-4314

and

Sonia Goel,
Department of Mathematics,
Maharishi Markandeshwar Engineering College,
Maharishi Markandeshwar (Deemed to be University),
Mullana (Ambala), Haryana, India
E-mail address: sonia.mangla14@gmail.com, ORCID: 0000-0003-1211-6003

and

Deepak Gupta,
Department of Mathematics,
Maharishi Markandeshwar Engineering College,
Maharishi Markandeshwar (Deemed to be University),
Mullana (Ambala), Haryana, India
E-mail address: guptadeepak@yahoo.co.in, ORCID: 0000-0002-9461-8770