ISSN: 0067-2904

A Hybrid Approach for Efficiently Solving Multi-Criteria Scheduling Problems on a Single Machine

Nagham Muosa Neamah [1,2]*, Bayda Atiya Kalaf [2]
[1] *Department of Mathematics, College of Science for Women, University of Baghdad, Baghdad, Iraq*
[2]*Department of Mathematics, College of Education for Pure Science Ibn-Al-Haitham, University of Baghdad, Baghdad, Iraq*

Received: 7/10/2023 Accepted: 19/11/2023 Published: xx

Abstract:
The presented study investigates the scheduling regarding n jobs on a single machine. Each n job is processed with no interruptions and becomes available for processing at time 0. The aim is to find a processing order with regard to jobs, minimizing "multi-criteria and multi-objective" for two problems. The first problem considers the summation completion time $\sum C_j$, summation late work $\sum V_j$, and maximal tardiness E_{max}, while the second problem considers the total completion time $\sum C_j$, total earliness $\sum E_j$, and maximum tardiness T_{max}. In addition, a sub-problem is presented for each problem and denoted by 1// $\sum C_j + \sum V_j + E_{max}$ and 1// $\sum C_j + \sum E_j + T_{max}$ for the first and second problems, respectively, which is an NP-hard problem. Two meta-heuristics methods "particle swarm optimization (PSO) and bee algorithm (BA)" were applied to acquire the optimal or near-optimal solution. Meta-heuristics methods solve problems of up to $n = 4000$ jobs. Finally, in an attempt to increase the overall search efficiency, a hybrid algorithm was created by combining two algorithms, a hybrid between BA and PSO was created to create an alternative search method that incorporates the best properties that each method offers during problem-solving. Moreover, by comparing the performance of local search methods with a hybrid strategy, the hybrid strategy method outperforms BA and PSO. In addition, it can solve problems up to $n = 8000$ jobs. Arithmetic results are calculated by coding (programming) algorithms using (MATLAB 2019a).

keywords: Single Machine Scheduling Problem (SMSP), Meta-Heuristics methods (MH), Multi-Criteria (MC), Multi-Objective (MO).

تقنية هجينة جديدة لحل مشاكل الجدولة متعددة المعايير على جهاز واحد

نغم موسى نعمه *[1و2], بيداء عطية [2]
[1]القسم الرياضيات, الكلية العلوم للبنات, الجامعة بغداد, بغداد, العراق
[2]القسم الرياضيات, الكلية التربية للعلوم الصرفة ابن الهيثم, الجامعة بغداد, بغداد, العراق

* Email: nagham2182011@gmail.com

الخلاصة:

بحثت الدراسة المقدمة في الجدولة المتعلقة بالوظائف n على جهاز واحد. ستتم معالجة كل مهمة بدون انقطاع وستصبح متاحة للمعالجة في الوقت 0. الهدف هو العثور على أمر معالجة فيما يتعلق بالوظائف، مما يقلل من المعايير المتعددة والأهداف المتعددة لمشكلتين. المشكلة الأولى، إجمالي وقت الإنجاز $\sum C_j$ ، وإجمالي العمل المتأخر $\sum V_j$ ، والحد الأقصى للتأخير E_{max} ، والمشكلة الثانية، إجمالي وقت الإنجاز $\sum C_j$ ، وإجمالي التبكير $\sum E_j$ ، والحد الأقصى للتأخير T_{max} بالإضافة إلى ذلك، تم تقديم مشكلة فرعية لكل مشكلة يرمز لها بالرمز $1// \sum C_j + \sum E_j + T_{max}$ و $1 // \sum C_j + \sum V_j + E_{max}$ للمسألتين الأولى والثانية على التوالي. وهي مشاكل NP−hard .تم استخدام طريقتين للبحث المحلي (تحسين سرب الجسيمات وخوارزمية النحل) للحصول على الحل الأمثل أو شبه الأمثل. تحل طرق البحث المحلية مشاكل تصل إلى $n = 4000$ وظيفة. وأخيرا، في محاولة لزيادة كفاءة البحث الشاملة، تم إنشاء خوارزمية هجينة من خلال الجمع بين خوارزميتين، تم إنشاء خوارزمية هجينة بين BA وPSO لإنشاء طريقة بحث بديلة تتضمن أفضل الخصائص التي تقدمها كل طريقة أثناء حل المشكلات. علاوة على ذلك، من خلال مقارنة أداء طرق البحث المحلية مع الاستراتيجية الهجينة، تتفوق طريقة الاستراتيجية الهجينة على BA وPSO، وبالإضافة إلى ذلك، يمكنها حل مشاكل تصل إلى $n = 8000$ وظيفة. يتم حساب النتائج الحسابية عن طريق خوارزميات الترميز (البرمجة) باستخدام MATLAB 2019a .

1. Introduction

The Machine Scheduling Problem (MSP), a crucial area of Operation Research (OR), and it can be defined as the problem of scheduling given n jobs, each of them requires one or more operations, on one or more machines during a given period of time in a way that minimizes a given objective function. Scheduling is a decision-making method that plays a major role in our daily life and it is used in information processing, production, purchasing, distribution and transportation, among other areas of manufacturing and service industries as well as in some military problems [1]. Mathematical techniques and heuristic strategies are used to allocate limited resources in order to obtain optimal or near-optimal solutions to the job scheduling problem used in companies. Local search heuristics are based on observations of processes in the physical and biological sciences [2][3]. The Machine Scheduling problem MSP has become significantly more complex and large-scale in recent decades as a result of the development of more complex modeling techniques with making more assumptions [4]. Meta-heuristic Optimization strategies have become increasingly popular in the research community as a means of solving huge, difficult problems [5]. This is mostly caused by the time-consuming nature and general unsuitability of classical procedures. For the purpose of solving MSP, meta-heuristic algorithms were created [6].

In 2017 [3], three local search algorithms (descent method, simulated annealing, and tabu search) are used to minimize multi-objective function $\left(\sum C_j + \sum T_j + T_{max} + E_{max}\right)$ up to 2000. In addition, the branch and bound method was used as exact method to solve this problem up to $n \leq 25$. In 2020 [7] used local search methods (LSM), simulated annealing, and particle swarm optimization to minimize multi-criteria and multi-objective function $\left(\sum C_j, \sum E_j\right)$, $\left(\sum C_j + \sum E_j\right)$ for n ≤ 1000. In 2020 [8], two local search methods (Bees algorithm and Particle Swarm Optimization) were used to minimize $\left(\sum C_j, R_L, T_{max}\right)$ and $\left(\sum C_j + R_L + T_{max}\right)$ up to n $\leq$ 1000. In 2020 [9] used simulated annealing, tabu search to solve the problem 1/ r_j / $T_{max} + \sum w_j + \sum w_j e^{-rC_j}$. Then, in 2022 [10], used two local search algorithms (genetic algorithm and particle swarm optimization) to minimize a multi-objective function, $\sum_{j=1}^{n}\left(C_j + T_j + E_j + V_j\right)$. In 2020 [11] used a heuristic algorithm to minimize the sum of total completion time, maximum earliness, and maximum tardiness in a single-machine scheduling. In 2023 [12], two local search methods simulated annulling and bees algorithm were proposed to minimize multi-

criteria $(\sum C_j, \sum V_j)$. In the same year,[13] three new hybrid algorithms hybridizing a new method with a genetic algorithm, taboo search, and simulated annealing, denoted as GA^H, TS^H, and SA^H are proposed to minimize the total penalty cost defined as the sum of tardy, early, and overtime costs.

Despite the fact that complicated real-world MSP problems have been successfully tackled using meta-heuristic algorithms, not all issues using the no-free-lunch theorem have a standard algorithm (SA)[6] . As a result, contemporary ideas of modifying self-adaptive algorithms or hybrid algorithms that facilitate the selection of the appropriate algorithm seek to overcome the implicit limitations of Meta-heuristics in dealing with actual MSP situations. Due to the use of hybrid algorithms in the current study, proposed a new hybrid algorithm that combines the PSO and the BA. A hybrid technique between "particle swarm optimization and bee algorithm" is proposed, by combining PSO and BA to improve the results of BA and PSO. Moreover, it is used for solving the problem up to $n > 4000$ jobs.

The rest of the paper is organized as follows: The conceptual framework is described in section 2. In the third Section, the mathematical formulation of the problems was presented. The fourth section presented the Hybrid BA-PSO algorithm, in addition to computational experiments for applications of PSO, BA, and BA-PSO. It is also presented comparison results for applying all problem-solving methods. The practical results of the proposed problems are evaluated in Section five. Finally, the main conclusions and further works are presented in Section six.

2. Conceptual framework

- The problems are organized and designed as multi-criteria mathematical models and a multi-objective sub-problem of the original problem is proposed.
- Two Meta-heuristics methods PSO and BA are proposed to solve these problems up to $n = 4000$.
- Moreover, a hybrid technique between "particle swarm optimization and bee algorithm" is proposed, by combining PSO and BA, which is adopted to find efficient solutions to this problem in a reasonable time up to $n = 8000$ jobs.

3. Mathematical Models

This problem can be illustrated on a single machine to schedule jobs to minimize multi-criteria and multi-objectives: At time 0, the number of available jobs is represented as $N = \{1,2, \dots, n\}$, (i. e. , $r_j = 0$ for all j) and they need processing on just one machine. For each j job, it has a processing time p_j, a due date d_j. In addition, a list of given jobs in the sequence $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$. This research article will use several terms that can be defined as follows:

BA: Bees Algorithm.
PSO: Particle Swarm Optimization
L_j: Lateness time of job j, s. t. $L_j = C_j - d_j$.
T_j: Tardiness for job j, s. t. $T_j = max\{0, L_j\}$, where L_j: Lateness time of job j, s. t. $L_j = C_j - d_j$. Aslo, T_{max}: Maximal tardiness s. t. $T_{max} = max_{j \in N}\{T_j\}$.
V_j: Late work s.t., $V_j = min\{T_j, p_j\} = min\{C_j - d_j, p_j\}$, $\sum V_j$: Total Late work.
E_j: Earliness time for job j, s. t. $E_j = max\{0, -L_j\}$, $\sum E_j$: Total earliness time.
C_j: Completion time for job j, where $C_j = \sum_{k=1}^{j} p_k$, $\sum C_j$: Total completion time.
F_{CVE}: Objective Function of the problem $(T_{CV}M_E)$, s.t., $F_{CVE} = Min(\sum C_j, \sum V_j, E_{max})$.
F_{SP} : Objective Function of sub-problem.

F_{CET}: Objective Function of the problem $(S_{CE}M_T)$, s.t., $F_{CET} = Min(\sum C_j, \sum E_j, T_{max})$
MOF: Multi-Objective Function.
MCF: Multi- Criteria Function.
EFSO*:* Efficient Solution [4], a schedule α^* is known as efficient solution or Pareto optimal or (non-dominated), if we cannot find another schedule α satisfying $h_j(\alpha) \leq h_j(\alpha^*), j = 1,2,..,n$, with at least one of the above considered a strict disparity. Another way is that α^* is dominated by α [11].
OP: optimal**,** σ^* is the schedule which is considered as optimal in the case when there is no other schedule σ that satisfies $f_j(\sigma) \leq f_j(\dot{\sigma}), j = 1{:}\, k$ (k: criteria's number), assuming a strict inequality for a minimum of one of the conditions that have been mentioned earlier. If not, σ can be considered dominant over $\dot{\sigma}$ [4].

Finding a schedule $\beta \in S$ is the aim of the first problem (S is the set of all possible feasible schedules that satisfies all the constraints of the problem) that minimizes the multi-criteria $F_{CVE} = min(\sum C_j, \sum V_j, E_{max})$. This problem denoted by $(T_{CV}M_E)$, and this problem is denoted by:

$$\left.\begin{array}{l} F_{CVE} = Min(\sum C_j, \sum V_j, E_{max}) \\ \text{subject to} \\ \left.\begin{array}{l} C_j \geq p_j(\beta), \\ C_j = \sum_{k=1}^{j-1} p_k(\beta) + p_j(\beta), \\ T_j \geq C_j - d_j(\beta), \\ E_j \geq d_j(\beta) - C_j, \\ V_j = min\{T_j, p_j(\beta)\}, \\ V_j \geq 0, E_j \geq 0, \text{and } T_j \geq 0, \end{array}\right\} j \text{ from 1 to } n \end{array}\right\} \quad (1)$$

For the problem$(T_{CV}M_E)$, sub-problem can be concluded: The $1//(\sum C_j + \sum V_j + E_{max})$ problem is referred to as the problem $(SP1)$, and it can be defined as follows*:*

$$\left.\begin{array}{ll} F_{SP} = Min(\sum C_j + \sum V_j + E_{max}) & \\ \text{subject to} & \\ C_j = \sum_{k=1}^{j} p_{\beta_k} & j = 1{:}\, n \\ C_j = C_{(-1+j)} + p_{\beta_j} & j = 2{:}\, n \\ E_j \geq d_{\beta_j} - C_j & j = 1{:}\, n \\ T_j \geq C_j - d_{\sigma_j} & j = 1{:}\, n \\ V_j = min\{T_j, p_{\beta_j}\} & j = 1{:}\, n \\ V_j \geq 0,\ E_j \geq 0,\ T_j \geq 0 & j = 1{:}\, n \end{array}\right\} \quad (2).$$

Finding the order of processing jobs that will minimize this sub-problem and it is the main goal. The objective of the sub-problem is to find the sequence of job processing that will minimize $\sum C_j + \sum V_j + E_{max}$ [1].
The second problem is denoted by $(S_{CE}M_T)$, and it can be formulated mathematically as follows:

$$\left.\begin{array}{ll} F_{CET} = Min\left(\sum C_j, \sum E_j, T_{max}\right) & \\ \text{subject to} & \\ C_1 = p_{\beta_1} & \\ C_j \geq p_{\beta_j} & j = 1{:}n \\ C_j = C_{\beta_{(j-1)}} + p_{\beta_j} & j = 2{:}n \\ T_j \geq C_j - d_{\beta_j} & j = 1{:}n \\ E_j \geq d_{\beta_j} - C_j & j = 1{:}n \\ T_j \geq 0, E_j \geq 0 & j = 1{:}n \end{array}\right\} \quad (3).$$

Furthermore, from the problem $(S_{CE}M_T)$ a sub-problem can be concluded: The $1//\left(\sum C_j + \sum E_j + T_{max}\right)$ problem is referred to as the problem $(SP2)$, and it can be defined as follows*:*

$$\left.\begin{array}{ll} F_{CET} = Min\left\{\sum C_j + \sum E_j + T_{max}\right\} & \\ \text{s. t.} & \\ C_1 = p_{\beta_1} & \\ C_j \geq p_{\beta_j} & j = 1{:}n \\ C_j = C_{\beta_{(-1+j)}} + p_{\beta_j} & j = 2{:}n \\ T_j \geq C_j - d_{\beta_j} & j = 1{:}n \\ E_j \geq d_{\beta_j} - C_j & j = 1{:}n \\ T_j \geq 0, E_j \geq 0 & j = 1{:}n \end{array}\right\} \quad (4).$$

4. Hybrid Meta-Heuristics Algorithm (BA-PSO)

The problems $T_{CV}M_E$ and $S_{CE}M_T$ in scheduling machine problems belong to the NP-hard meta-heuristics category [14], and they are considered large-scale problems (containing a large number of jobs) and they cannot be solved by analytical methods. Meta-Heuristics works well for large-scale MSP challenges. On the other hand, hybridization leads to powerful (robust) solution methods. Modern Meta-heuristics take into account intensification and diversification (exploration and exploitation), while the building of the hybrid model, which consists of these two basic elements. For an algorithm to be efficient and successful (both in terms of speed and quality), it must be able to effectively scan the entire search space and condense its search into the area of the optimal or near-optimal solution. To improve the speed and quality of any algorithm, a balance must be struck between exploration and exploitation. An effective combination of these two basic approaches almost guarantees global optimum [6]. Search algorithms were chosen to balance exploration and exploitation. In this paper, we propose a new hybrid algorithm that combines the PSO algorithm and the BA algorithm, to solve the models $T_{CV}M_E$ and $S_{CE}M_T$, by combining PSO as well as BA. This method aims to create an alternative research method that incorporates the best properties offered by each method while solving problems. PSO is a very simple idea that does not require a complex data structure to implement. It does not use any complex or expensive mathematical functions. Moreover, it does not require a lot of memory. Fast convergence, minimal control parameters, easy calculations, good performance, and the absence of derived calculations make PSO an attractive option for solving problems. While, the bee algorithm is more scalable, when it comes to finding and collecting food. Food processors are more efficient because they require fewer steps.
First, we introduce the steps of the PSO algorithm as follows:

Algorithm 1: PSO Algorithm

Step 1: Dimension (d), LB(minx), UB(maxx), No. of Particles (N), Maximum number of iterations (max_iter).
Step 2: Randomly initialize Swarm (S) population of N particles Position (P) and velocity (V) (i=1,2,…, N).
Step 3: Select hyperparameter values w, c1 and c2.
Step 4: Calculate the Fitness value for each particle.
Step5: FOR Iter=1 : max_iter { *loop max_iter times* }
FOR i=1: N { *for each particle* }
a. Compute the new velocity of the ith particle (V)
S[i].V=w*S[i].V+r1*c1*(S[i].bestP - S[i].P)+
r2*c2*(bestP_S - S[i].P) ;
b. Compute the new position of ith particle using its new velocity
S[i].P = S[i].P + S[i].V;
c. **IF** position is not in the range [minx, maxx] then clip it
IF S[i].P < minx : S[i].P = minx ;
IF S[i].P > maxx : S[i].P = maxx ;
d. Calculate Fitness S[i]. Fit and update the new best of each particle and the new best of Swarm.
IF S[i].Fit < S[i].bestFit
S[i].bestFit = S[i].Fit ;
S[i].bestP = S[i].P ;
ENDIF
IF S[i].Fit < best_fitness_swarm
best_fitness_swarm = S[i].Fit ;
best_pos_swarm = S[i].P ;
ENDIF
END-FOR {i}
END-FOR {iter}
Step 6: Return the best particle of Swarm.

Secondly, we introduce the steps of the BA algorithm as follows:

Algorithm 2: BA Algorithm

Step 1: Dimension (d), No. of Bees (N), LB, UB, Maximum number of iterations (max_iter).
Step 2: Randomly initialize Swarm (S) population of N Bees X∈[LB,UB], (i=1,2,…,N);
Step 3: Select hyperparameter values m**,** e, nep, nsp, ngp.
Step 4: Calculate the Fitness value for each Bee.
Step 5: FOR Iter=1 : max_iter { *loop max_iter times* }
FOR i=1: e
Swap random neighborhood for S[i].X;
Calculate Fitness S[i].Fit;
END-FOR {i}
FOR i=e+1: m
Swap random neighborhood for S[i].X;
Calculate Fitness S[i].Fit;
END-FOR {i}
Randomly initialize Swarm (S) population of N-m Bees X (i=m+1,2,…, N).
SORT By **(**S.Fit**);**
IF S[i].Fit < best_fitness_Bee

```
            best_fitness_Bee = S[i].Fit ;
            best_Bee = S[i].X ;
          END-IF
        END-FOR {iter}
```
Step 6: Return the best Bee of Swarm.

Now, the hybridization of BA with PSO can be seen when applying equation (2) of the position that updated in the PSO algorithm in step (5) by applying the swapping of the neighborhood for m best bees in the swarm, which applied for BA in step (5). This hybrid will be more helpful in improving the best position for each particle in the swarm.
The steps of the BA-PSO algorithm are as follows:

Algorithm 3:BA-PSO Algorithm

Step 1: Dimension (d), LB(minx), UB(maxx), No. of Particles (Bees) (N), Maximum number of iterations (max_iter).
Step 2: Randomly initialize Swarm (S) population of N particles Position (P) (Bees $X \in [LB,UB]$) and velocity (V) (i=1,2,…,N).
Step 3: Select hyperparameter values w, c1, c2, m**,** e, nep, nsp, ngp.
Step 4: Calculate Fitness value for each particle (Bee).
Step 5: FOR Iter=1 : max_iter { *loop max_iter times* }
FOR i=1: N { *for each particle* (*Bee*) }
a. Compute new velocity of ith particle (V)

```
S[i].V=w*S[i].V+r1*c1*(S[i].bestP - S[i].P)+
r2*c2*( bestP_S - S[i].P) ;
```

b. Compute new position of ith particle using its new velocity

```
S[i].P = S[i].P + S[i].V;
```

c. **IF** position is not in range [minx, maxx] then clip it

```
IF S[i].P < minx : S[i].P = minx ;
IF S[i].P > maxx : S[i].P = maxx ;
```

d. Randomly initialize Swarm (S) population of N-m Particles (Bees) X (i=m+1,2,…,N).
SORT By **(**S.Fit**);**

```
IF S[i].Fit < best_fitness_Bee
    best_fitness_Bee = S[i].Fit ;
    best_Bee = S[i].X ;
END-IF
```

e. Calculate Fitness S[i].Fit and update new best of each particle and new best of Swarm.

```
IF S[i].Fit  < S[i].bestFit
      S[i].bestFit = S[i].Fit ;
      S[i].bestP = S[i].P ;
ENDIF
IF S[i].Fit < best_fitness_swarm
      best_fitness_swarm = S[i].Fit ;
      best_pos_swarm = S[i].P ;
ENDIF
END-FOR {i}
END-FOR {iter}
```
Step6: Return best particle of Swarm.

Note: The hybrid between BA and PSO is represented by adding the step (5-d) in the BA-PSO algorithm.

The BA-PSO results are compared with the results of PSO and BA meta-heuristics methods.

4.1 Computational Experiments

All algorithms were done in MATLAB R2019a and implemented on Intel(R) Core (TM) i7-2630 QM CPU @ 2.00 GHz 2.00 GHz and 4.00 GB of RAM. Local searches are run for a maximum of 10 minutes (600 seconds), and if the instance takes longer than 600 seconds, the instance will not be resolved and stopped.

4.2 Test Problem

The problems were generated randomly and for each j job, where $j \in N, N = \{1, \dots, n\}$.

- The processing time were distributed uniformly during the period $[1,10]$.
- The due date was uniformly distributed over the period $[1,70]$ s.t., $d_j \in \begin{cases} [1,30], & 1 \le n \le 29 \\ [1,40], & 30 \le n \le 99 \\ [1,50], & 100 \le n \le 999 \\ [1,70], & \text{otherwise} \end{cases}$, under condition $d_j \ge p_j \ \forall\, j = 1, \dots, n$.
- All results of applying all proposed methods represent averages of the results of (5) examples for each n.
- After averaging 10,000 cycles or more using the 10-8-1 configuration, training is considered finished.

The used abbreviations are:

Abbreviations	Description
ACT/S	Average processing time (Average of CPU-Time per second).
ANEFS	Average number of efficient solutions.
Av	Average value of objective function.
EX	Example Number.
n_i	The jobs number, while i denoted the number of problems tested.
RL	0 < Real < 1

The parameters in meta-heuristics methods were used as follows:

For PSO: Particle's count ($N\_Par = 20$), maximal velocity [v_{max} = number of available jobs], minimal velocity ($v_{min} = 1$), weight of inertial ($w \in [0.4,0.9]$). The first acceleration parameter ($c_1 \in [0.5,2]$), the second acceleration parameter ($c_2 = c_1 = 2$), the population conservation diversity (random $r_1, r_2 \in [0,1]$) and some hundreds of generations.

For BA: Scout bee population (Count of scout bees) ($nn = 20$, number of available Jobs (n)), the number of lections selected from among the n sites visited ($m = 5$), the number of sites that are better than the mm of sites selected ($e = 2$). For the best sites, the number of bees assigned ($nep = 5$), at the other selected sites, the number of bees was recruited $(m - e)(nsp = 3)$, and a maximum number iteration ($MI = 1000$).

■ Applying PSO and BA to the first problem $1//\left(\sum C_j, \sum V_j, E_{max}\right)$ and sub-problem $1// \sum C_j + \sum V_j + E_{max}$ for different numbers of jobs, and comparing efficient results between PSO and BA are shown in the Table 1.

Table 1: PSO and BA comparison outcomes for the first problem 1//$(\sum C_j, \sum V_j, E_{max})$ and sub-problem 1// $\sum C_j + \sum V_j + E_{max}$ for different numbers of jobs.

EX	PSO($T_{CV}M_E$)		BA($T_{CV}M_E$)		PSO(SP)		BA(SP)	
	MCF	TIME	MCF	TIME	MOF	TIME	MOF	TIME
n_5	$AV(F_{CVE})$	ACT/S	$AV(F_{CVE})$	ACT/S	$AV(F_{SP1})$	ACT/S	$AV(F_{SP1})$	ACT/S
10	(241.5,32.1,17.8)	RL	(279.2,33.7,19.2)	RL	287.2	RL	298.6	RL
40	(3489.8,201.7,15.5)	1.3	(4464.4,204.7,15.7)	RL	3557.4	1.4	4458.6	RL
70	(10922,377,20)	2.4	(13143,359,16)	RL	10825.2	3.7	13105.4	RL
100	(22285,546,21)	3.0	(27779,547,16)	1.7	22545.0	2.7	27794.2	1.1
400	(374360 ,2180,20)	5.5	(433010,2180,20)	6.2	376577.0	4.2	429908.8	5.6
700	(1180958.8,3848.2,18.9)	88.8	(1344027.4,3848.2,14.2)	153.3	1149249.0	83.7	1334630.2	73.7
1000	(2453939.2,5490.7,16.3)	97.3	(2739034.8,5492.9,15.2)	266.7	2421338.6	8.1	2721013.4	98.2
2000	(9951740.6,10961.7,13.2)	221.0	(10923270.3,10963.6,16.7)	316.7	9907230.0	19.0	10865678.2	108.2
3000	(22934199.2,16420.1,16.9)	257.2	(24585762.6,16420.0,12.8)	430.9	22923685.8	161.4	24398313.4	168.4
4000	(41063190.9,21881.9,18.5)	1168.834	-	-	40546664.0	229.468	-	-

■ Applying PSO and BA to the second problem 1//$(\sum C_j, \sum E_j, T_{max})$and sub-problem 1// $\sum C_j + \sum E_j + T_{max}$ for different numbers of jobs, and comparing the results' efficiency between PSO and BA as it is shown in Table 2.

Table 2: PSO and BA comparison results for the first problem 1//$(\sum C_j, \sum E_j, T_{max})$ and sub-problem 1// $\sum C_j + \sum E_j + T_{max}$ for different numbers of jobs.

EX	PSO($S_{CE}M_T$)		BA($S_{CE}M_T$)		PSO(SP)		BA(SP)	
	MOF	TIME	MOF	TIME	MOF	TIME	MOF	TIME
n_5	$AV(F_{CET})$	ACT/S	$AV(F_{CET})$	ACT/S	$AV(F_{SP2})$	ACT/S	$AV(F_{SP2})$	ACT/S
4	(60.8, 24.2,2.2)	RL	(61.1,24.8,3.2)	RL	84.2	RL	84.2	RL
10	(256.0,24.4,32.1)	RL	(287.3,22.9,37.3)	RL	308.0	RL	322.4	RL
40	(3536.9,22.2,203.3)	1.2	(4440.9,20.1,208.5)	RL	3633.4	1.0	4518.6	RL
70	(9613.5,23.3,338.6)	1.9	(13439,16,374)	RL	9664.4	1.9	13402.8	RL
100	(22950,28,550)	2.8	(27992,14,551)	1.3	22630.0	2.5	27659.4	1.0
400	(374420,10,2190)	4.3	(437160,20,2190)	6.7	367993.2	6.1	432600.8	5.7
700	(1140294.9,18.3,3825.5)	269.9	(1357740.5,14.2,3906.2)	233.5	1182448.0	3.8	1348859.8	25.4
1000	(2440929.0,17.2,5493.9)	142.8	(2734789.4,13.4,5497.9)	395.8	2333107.2	124.2	2709013.6	79.8
2000	(9942739.2,19.2,10970.4)	252.5	(10949480.9,19.0,10972.6)	265.6	9919191.0	437.1	10858719.2	132.5
3000	(22747664.3,20.5,16428.6)	32.3	(24546895.3,13.0,16429.2)	727.3	22793982.0	157.3	24410788.4	116.1
4000	(40701320.7,21882.5,14.1)	842.3	-	-				

■ The efficient results of BA − PSO are compared with those of BA and PSO for the first problem 1//$(\sum C_j, \sum V_j, E_{max})$ for different numbers of jobs, as it is displayed in Table 3.

Table 3: Results of applying BA − PSO and comparison with BA, PSO for the first problem 1//$(\sum C_j, \sum V_j, E_{max})$ for different numbers of jobs.

EX	BA − PSO($T_{CV}M_E$)		PSO($T_{CV}M_E$)		BA($T_{CV}M_E$)	
	MCF	TIME	MCF	TIME	MCF	TIME
n_5	$AV(F_{CVE})$	ACT/S	$AV(F_{CVE})$	ACT/S	$AV(F_{CVE})$	ACT/S
10	(189.9,19.2,20.9)	RL	(241.5,32.1,17.8)	RL	(279.2,33.7,19.2)	RL
40	(3466.5,201.4,17.6)	1.9	(3452.8,200.9,18.9)	1.4	(4464.4,204.7,15.7)	RL

70	(10678.0,350.8,16.3)	3.7	(10922,377,20)	2.4	(13143,359,16)	RL
100	(22939.5,545.4,17.9)	5.5	(22285,546,21)	3.0	(27779,547,16)	1.7
400	(370071.8,2180.4,19.4)	23.0	(372160,2180,20)	5.0	(433010,2180,20)	6.2
700	(1124860.4,3811.5,18.4)	35.2	(1180958.8,3848.2,18.9)	88.8	(1344027.4,3848.2,14.2)	153.3
1000	(2295791.9,5489.4,17.9)	59.4	(2453939.2,5490.7,16.3)	97.3	(2739034.8,5492.9,15.2)	266.7
2000	(9475283.7,10917.9,16.9)	101.7	(9951740.6,10961.7,13.2)	221.0	(10923270.3,10963.6,16.7)	316.7
3000	(21869964.4,16377.9,15.8)	150.6	(22934199.2,16420.1,16.9)	257.2	(24585762.6,16420.0,12.8)	430.9
4000	(39792541.3,22087.9,18.6)	217.9	(41063190.9,21881.9,18.5)	1168.8	-	-
5000	(62652747.8,27452.5,19.5)	270.1	-	-	-	-
6000	(90710011.2,32945.9,18.8)	318.0	-	-	-	-
7000	(124816483.2,38442.5,18.3)	383.5	-	-	-	-

■ The optimal results of BA − PSO are compared with those of BA and PSO for the first problem 1// $\sum C_j + \sum V_j + E_{max}$ for different numbers of jobs, Table 3 displays these outcomes.

Table 4: Results from using PSO − BA and comparing them to BA and PSO for the problem 1// $\sum C_j + \sum V_j + E_{max}$ for various numbers of jobs.

EX	BA − PSO($SP1$)		PSO($SP1$)		BA($SP1$)	
	MOF	TIME	MOF	TIME	MOF	TIME
n_5	$AV(F_{SP1})$	ACT/S	$AV(F_{SP1})$	ACT/S	$AV(F_{SP1})$	ACT/S
10	224.2	RL	287.2	RL	298.6	RL
40	3622.2	1.8	3557.4	1.4	4458.6	RL
60	7572.0	2.5	7514.0	1.7	9444.0	RL
70	10353.0	3.7	10825.2	3.7	13105.4	RL
100	22398.8	5.6	22545.0	2.7	27794.2	1.1
400	342926.6	21.1	376577.0	4.2	429908.8	5.6
700	1069815.2	31.0	1149249.0	83.7	1334630.2	73.7
1000	2272494.4	44.7	2421338.6	8.1	2721013.4	98.2
2000	9290653.2	95.2	9907230.0	19.0	10865678.2	108.2
3000	21606031.2	144.2	22923685.8	161.4	24398313.4	168.4
4000	39878241.2	194.8	40546664.0	229.4	-	-
5000	62379935.6	349.5	-	-	-	-
6000	90475774.6	309.1	-	-	-	-
7000	123836583.2	335.0	-	-	-	-
8000	162636634.6	376.7	-	-	-	-

■ The efficient results of BA − PSO are compared with those of BA and PSO for the first problem 1//$\left(\sum C_j, \sum E_j, T_{max}\right)$ for different numbers of jobs, and these results are shown in Table 5.

Table 5: The results of applying BA − PSO and comparison findings for the second problem 1//$\left(\sum C_j, \sum E_j, T_{max}\right)$ using BA and PSO for different numbers of jobs.

EX	BA − PSO($S_{CE}M_T$)		PSO($S_{CE}M_T$)		BA($S_{CE}M_T$)	
	MCF	TIME	MCF	TIME	MCF	TIME
n_5	$AV(F_{CET})$	ACT/S	$AV(F_{CET})$	ACT/S	$AV(F_{CET})$	ACT/S

10	(192.0,18.8,21.5)	RL	(256.0,24.4,32.1)	RL	(287.3,22.9,37.3)	RL
40	(3535.8,200.4,16.3)	1.9	(3536.9,22.2,203.3)	1.2	(4440.9,20.1,208.5)	RL
70	(10523.1,350.1,18.7)	3.7	(9613.5,23.3,338.6)	1.9	(13439,16,374)	R
100	(22996.5,546.7,17.9)	5.0	(22950,28,550)	2.8	(27992,14,551)	1.3
400	(360353.7,2180.6,18.4)	21.0	(374420,10,2190)	4.3	(437160,20,2190)	6.7
700	(1107115.0,3811.1,20.5)	34.2	(1140294.9,18.3,3825.5)	269.9	(1357740.5,14.2,3906.2)	233.5
1000	(2366517.2,5489.9,17.9)	48.9	(2440929.0,17.2,5493.9)	142.8	(2734789.4,13.4,5497.9)	395.8
2000	(9473947.5,10917.8,17.6)	103.6	(9942739.2,19.2,10970.4)	252.5	(10949480.9,19.0,10972.6)	265.6
3000	(22022993.5,16376.5,16.8)	151.6	(22747664.3,20.5,16428.6)	32.3	(24546895.3,13.0,16429.2)	727.3
4000	(39953660.9,22086.2,16.0)	215.2	(40701320.7,21882.5,14.1)	842.3	-	-
5000	(63148609.3,27451.9,23.0)	276.6	-	-	-	-
6000	(90710011.2,32945.9,18.8)	729.6	-	-	-	-
7000	(124024405.6,38444.8,16.9)	377.8	-	-	-	-

■ The optimal results of $\mathrm{BA}-\mathrm{PSO}$ are compared with those of BA and PSO for the first problem 1// $\sum C_j + \sum E_j + T_{max}$ for different numbers of jobs, and Table 6 displays these results.

Table 6: Application results of $\mathrm{BA}-\mathrm{PSO}$ and comparison with the results of PSO and BA for the 1// $\sum C_j + \sum E_j + T_{max}$ problem for different numbers of jobs.

EX	$\mathrm{BA}-\mathrm{PSO}(SP2)$		$\mathrm{PSO}(SP2)$		$\mathrm{BA}(SP2)$	
	MOF	TIME	MOF	TIME	MCF	TIME
n_5	$AV(F_{SP2})$	ACT/S	$AV(F_{SP2})$	ACT/S	$AV(F_{SP2})$	ACT/S
10	225.8	RL	308.0	RL	322.4	RL
40	3606.4	2.0	3633.4	1.0	4518.6	RL
70	10312.0	3.5	9664.4	1.9	13402.8	RL
100	22526.2	5.4	22630.0	2.5	27659.4	1.0
400	346594.8	20.5	367993.2	6.1	432600.8	5.7
700	1063224.0	32.0	1182448.0	3.8	1348859.8	25.4
1000	2263702.8	45.9	2333107.2	124.2	2709013.6	79.8
2000	9338328.0	91.6	9919191.0	437.1	10858719.2	132.5
3000	21708195.6	166.8	22793982.0	157.3	24410788.4	116.1
4000	39702542.0	182.7	40592615.6	185.5	-	-
5000	62463734.0	388.2	65009326.8	32.9	-	-
6000	62463734.0	388.2	-	-	-	-
7000	90487941.2	283.3	-	-	-	-
8000	124004991.8	336.4	-	-	-	-

5. Evaluate the practical results of the proposed problems

The results presented in this section are based on computational experiments and they include:

- From Tables 1 and 2, we note that these results show that the value averages for using PSO (F_{CVE}) are better than BA (F_{CVE}), for problems $T_{CV}M_E$ and sub-problem for $T_{CV}M_E$, for different n. Also, the average processing time in MATLAB (CPU time) for PSO (F_{CVE}) is less than that of BA (F_{CVE}).
- From Tables 1 and 2, we note that these results show that the value averages for using PSO (F_{CET}) are better than BA (F_{CET}), for problems $S_{CE}M_T$ and sub-problem for $S_{CE}M_T$, for different n. Also, the average processing time in MATLAB (CPU time) for PSO (F_{CET}) is less than that of BA (F_{CET}).

- From Tables 3 and 4, we note that these results show that the value averages for using BA-PSO (F_{CVE}) are better than PSO (F_{CVE}) and BA (F_{CVE}), for problems $T_{CV}M_E$ and sub-problem for $T_{CV}M_E$, for different n. Also, the average processing time in MATLAB (CPU time) for PSO (F_{CVE}) is less than that of BA-PSO (F_{CVE}) and BA (F_{CVE}).
- From Tables 5 and 6, note these results show that the value averages for using BA-PSO (F_{CET}) are better than PSO (F_{CET}) and BA (F_{CVE}), for problems $S_{CE}M_T$ and sub-problem for $S_{CE}M_T$, for different n. Also, the average processing time in MATLAB (CPU time) for PSO (F_{CET}) is less than that of BA-PSO (F_{CET}) and BA (F_{CET}).

6. Discussions and Conclusions

In the current study, two meta-heuristics methods PSO and BA were applied to solve two problems on single-machine scheduling; $1//(\sum C_j, \sum V_j, E_{max})$, $1//(\sum C_j, \sum E_j, T_{max})$ which are denoted by $T_{CV}M_E, S_{CE}M_T$ respectively, and for sub-problem is derived for each problem to find the best or closest to best solution up to $n = 4000$ jobs. Finally, a hybrid between PSO and BA is done to create an alternative search method that incorporates the best properties that each method offers during problem-solving. Moreover, by comparing the performance of meta-heuristics methods with a hybrid strategy, the hybrid strategies method outperforms other methods up to $n = 8000$ jobs. Finally, the results of BA-PSO were better than that of BA and PSO, and the results of PSO were better than that of BA in all research problems.

Acknowledgements: The authors of this paper would like to thank the referees for their valuable suggestions and helpful comments

Compliance with ethical standards

Conflicts of interest the authors declare that they have no conflicts of interest.

Reference

[1] N. M. Neamah and B. A. Kalaf, "Solving the multi-criteria: total completion time, total late work, and maximum earliness problem," *Period. Eng. Nat. Sci.*, vol. 11, no. 3, pp. 46–57, 2023, doi: 10.21533/pen.v11i3.3559.g1288.

[2] B. A. Amin and A. M. Ramadan, "Novel Heuristic Approach for Solving Multi-objective Scheduling Problems," *Ibn AL- Haitham J. Pure Appl. Sci.*, vol. 34, no. 3, pp. 50–59, 2021, doi: 10.30526/34.3.2677.

[3] T. S. A. Razaq and H. M. Motair, "Solving Composite Multi objective Single Machine Scheduling Problem Using Branch and Bound and Local Search Algorithms," *Al-Mustansiriyah J. Sci.*, vol. 28, no. 3, 2017, doi: http://doi.org/10.23851/mjs.v28i3.122 Solving.

[4] A. A. Jawad, F. H. Ali, and W. S. Hasanain, "Using heuristic and branch and bound methods to solve a multi-criteria machine scheduling problem," *Iraqi J. Sci.*, vol. 61, no. 8, pp. 2055–2069, 2020, doi: 10.24996/ijs.2020.61.8.21.

[5] B. Atiya, A. J. K. Bakheet, I. T. Abbas, M. R. A. Bakar, L. L. Soon, and M. Bin Monsi, "Application of simulated annealing to solve multi-objectives for aggregate production planning," *AIP Conf. Proc.*, vol. 1739, no. November, 2016, doi: 10.1063/1.4952566.

[6] A. A. Zaidan, B. Atiya, M. R. Abu Bakar, and B. B. Zaidan, "A new hybrid algorithm of simulated annealing and simplex downhill for solving multiple-objective aggregate production planning on fuzzy environment," *Neural Comput. Appl.*, vol. 31, no. 6, pp. 1823–1834, 2019, doi: 10.1007/s00521-017-3159-5.

[7] F. H. Ali and A. A. Jawad, "Minimizing the total completion time and total earliness time functions for a machine scheduling problem using local search methods," *Iraqi J. Sci.*, vol. 2020, pp. 126–133, 2020, doi: 10.24996/ijs.2020.SI.1.17.

[8] F. H. Ali and M. G. Ahmed, "Local Search Methods for Solving Total Completion Times, Range of Lateness and Maximum Tardiness Problem," *Proc. 6th Int. Eng. Conf. ' 'Sustainable Technol. Dev. IEC 2020*, pp. 103–108, doi: 10.1109/IEC49899.2020.9122821.

[9] T. Jabbar Khraibet, W. Khalid Jaber, and L. Abed Dawood, "Local Search Methods to Find

Approximate Solution for the Sum of Two Criteria with Unequal Ready Times in Machine Scheduling Problem," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 928, no. 4, 2020, doi: 10.1088/1757-899X/928/4/042013.

[10] A. S. Hameed and Hanan Ali Chachan, "Genetic Algorithm and Particle Swarm Optimization Techniques for Solving Multi-Objectives on Single Machine Scheduling Problem," *Ibn AL-Haitham J. Pure Appl. Sci.*, vol. 33, no. 1, p. 119, 2020, doi: 10.30526/33.1.2378.

[11] D. A. Hassan, N. Mehdavi-Amiri, and A. M. Ramadan, "A heuristic approach to minimize three criteria using efficient solutions," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 6, no. 1, pp. 334–341, 2022, doi: 10.11591/ijeecs.v26.i1.pp334-341.

[12] F. H. Ali, R. J. Mitlif, and W. A. Hussein, "Exact and Near Pareto Optimal Solutions for Total Completion Time and Total Late Work Problem," *Iraqi J. Sci.*, vol. 64, no. 7, pp. 3485–3494, 2023, doi: 10.24996/ijs.2023.64.7.29.

[13] J. Latthawanichphan, W. Sukkerd, W. Songserm, and T. Wuttipornpun, "Novel Hybrid Algorithms for a Single Machine Scheduling Problem With an Overtime Constraint," *Int. J. Knowl. Syst. Sci.*, vol. 13, no. 1, pp. 1–26, 2023, doi: 10.4018/ijkss.298708.

[14] A. M. A. Hariri, C. N. Potts, and L. N. Van Wassenhove, "Single Machine Scheduling to Minimize Total Weighted Late Work," *ORSA J. Comput.*, vol. 7, no. 2, pp. 232–242, May 1995, doi: 10.1287/ijoc.7.2.232.