International Journal of Production Management and Engineering

https://doi.org/10.4995/ijpme.2023.18077

Received: 2022-07-13 Accepted: 2023-07-10

BASA: An Improved Hybrid Bees Algorithm for the single machine scheduling with early/tardy jobs

Ahmed Adnane Abdessemed [a1], **Leila Hayet Mouss** [a2], **Khaled Benaggoune** [a3]

[a] Laboratory of Automation and Production Engineering (LAP), Department of Industrial Engineering, University of Batna 2, 53, Constantine Road, Fesdis, Batna 05078, Algeria.

[a1] *adnane.abdessemed@univ-batna2.dz*, [a2] *h.mouss@univ-batna2.dz*, [a3] *k.benaggoune@univ-batna2.dz*

Abstract:

In this paper, we present a novel hybrid meta-heuristic by enhancing the Basic Bees Algorithm through the integration of a local search method namely Simulated Annealing and Variable Neighbourhood Search like algorithm. The resulted hybrid bees algorithm (BASA) is used to solve the Single Machine Scheduling Problem with Early/Tardy jobs, where the generated outcomes are compared against the Basic Bees Algorithm (BA), and against some stat-of-the-art meta-heuristics. Computational results reveal that our proposed framework outperforms the Basic Bees Algorithm, and demonstrates a competitive performance compared with some algorithms extracted from the literature.

Key words:

Bees Algorithm, Single Machine Scheduling, Early/Tardy, Simulated Annealing, meta-heuristic.

1. Introduction

The diversification of product requirements by customers is a typical feature of today's manufacturing environments, as well as the irregular arrival of orders and complex changes in manufacturing conditions. In order to adequately react to such perturbed environment, manufacturing is progressing towards an individual production or flexible manufacturing model, which characterizes the modern industry 4.0 (Lim et al., 2021). The ultimate form of production planning is to guarantee that all tasks are delivered precisely on time. However, it is practically difficult to meet the delivery requirements of the customers for all concerned tasks, due to many constraints, such as the uncertainty of the requests, and resource availability.

The single-machine scheduling with the objective of minimizing the total sum of earliness and tardiness penalties (SMSPET) is one of the challenging problems in which the objective is to combine both respecting tasks due dates and minimizing inventory costs, by considering both earliness and tardiness penalties. The SMSPET is known to be a strongly NP-hard problem (Wan and Yuan, 2013), and is characterised by a non-regular objective function making the resolution an intractable process in polynomial time (Yau et al., 2008).

The SMSPET denoted in Graham notation by $1||\Sigma(\alpha_i E_i+\beta_i T_i)$ can be defined as follows (Baker and Scudder, 1990): Let j be a set of n jobs to run on a single machine, neither idle time nor job preemption are allowed; all jobs are available at time zero. Each job i has four attributes: processing time P_i, due date d_i, earliness penalty α_i if the machine finishes the job before the due date, tardiness penalty β_i if the job is terminated after the due date. The objective is to find a processing sequence for the jobs in j that minimizes the total sum of earliness and tardiness penalties for all jobs. Let C_i be the completion time of the job i so the Earliness and the Tardiness

To cite this article: Abdessemed, A.A., Mouss, L.H., Benaggoune, K. (2023). BASA: An Improved Hybrid Bees Algorithm for the single machine scheduling with early/tardy jobs. *International Journal of Production Management and Engineering, 11*(2), 167-177. https://doi.org/10.4995/ijpme.2023.18077

expressions can be written as $E_i=max\{d_i - C_i, 0\}$, and $T_i = max\{C_i - d_i, 0\}$, respectively. Then, the objective function is given by:

$$f(x) = min \sum_{i=1}^{n} (\alpha_i E_i + \beta_i T_i) \quad (1)$$

Various approaches of dealing with the SMSPET have been suggested in the literature, ranging from exact to approximate methods. The earliest work on the SMSPET is presented by Abdul-Razaq and Potts (1988) in which a dynamic programming procedure for the problem is developed. Dynamic programming was also used to solve the SMSPET in the work of Tanaka et al. (2009), this later introduced a successive sublimation dynamic programming-based exact framework for the general single machine scheduling problem without idle time. Also branch and bound method was deployed for the SMSPET in Sourd and Kedad-Sidhoum (2008) and Sourd (2009). Exact methods ensure finding the optimal solution but from another hand they are computationally expensive and when subject to some constraints, the solution cannot be determined in a polynomial time.

Meta-heuristics are likely to find near-optimal solutions in a reasonable time. Many meta-heuristics are used to deal with the SMSPET in the literature. M'Hallah and Alhajraf in M'Hallah and Alhajraf (2016) presented three variants of Ant Colony systems to solve the SMSPET, namely Ant System using pairwise exchange as an exploration technique (ASD), Ant System with Simulated annealing (ASS), in which Simulated annealing is used to explore the search space, and Ant System with variable neighbourhood search (ASV), which uses the variable neighbourhood search algorithm to explore the search space around the ants. ASV is suggested as a best-performing algorithm among the ant systems presented. Also, Yurtkuran and Emel (2016) implemented a discrete version of the artificial bee colony (dABC) algorithm in which several modifications are achieved to adapt the original continuous Artificial Bee Colony (ABC) to the SMSPET. In the dABC, several search operators are investigated to generate neighbour solutions, also novel crossover operators are employed to improve the algorithm's exploration and exploitation behaviour. It was illustrated that the dABC yields better or comparable performances with respect to the Ant colony system algorithms presented in M'Hallah and Alhajraf (2016). Furthermore, the dABC exhibits higher efficiency versus other ABC approaches.

As dictated by the No Free Lunch theorem (NFL), described in Ho and Pepyne (2001), even if an algorithm **A** has obtained a high performance on a set of instances of an optimization problem, there certainly exists another algorithm **B** that performs better on other set of instances of the same problem. The NFL theorem encourages further research in trying to find a better algorithm than the existing ones.

Several optimization algorithms are inspired by the individual or social behaviour of animals and insects in nature. The Bees Algorithm (BA) is one of the modern optimization algorithms, it is a meta-heuristic that imitates the foraging behaviour of honey bees. The BA algorithm has a significant ability to explore a large solution space, and has been used to deal with many optimization problems with good performances. However, the BA algorithm has a number of drawbacks, including the low convergence rate when finding solutions (Yuce et al., 2017). In this paper, we propose an improved version of the bees algorithm by including the simulated annealing algorithm (SA) and the variable neighbourhood search algorithm as local search procedures within the same framework. The performance of the proposed hybrid approach will be evaluated and discussed on the basis of the Single machine scheduling problem with Early/Tardy jobs.

Therefore, the main contributions of this work are:

- Solving the single machine scheduling problem with early/tardy jobs to near-optimality;
- Proposing a hybridization scheme by combining Bees and Simulated Annealing strategies;
- Enhancing the Bees Algorithm behaviour by improving the local search phase.

The remainder of this paper is organized as follows. Section 2 emphasizes the background of the BA and its use in the literature. Section 3 is dedicated to the elaborated algorithm and the enhancements performed over the original Bees Algorithm. Section 4 reports the computational results on the SMSPET, and finally, some conclusions are recapitulated in Section 5.

2. Background of Bees Algorithm

In this section, some basic concepts related to the natural foraging behaviour of honey bees in addition to the associated original algorithm are depicted.

2.1. Honey bees foraging habits in the wild

Using clever exploitation and exploration strategies, a colony of honey bees may gain a broader food supply in large areas in the wild (Seeley, 2009). The process of foraging begins with scout bees searching for promising flower patches at random. When scout bees locate food supplies, they execute the "waggle dance" to alert the remainder of the colony about the quality, distance, and direction of the food source. This information will influence the number of recruited forager bees. The better the food source, the more bees are employed to search around it by performing a local search process. Scout bees would pursue a global random search, while some bees would be recruited to conduct a local search, to ensure that the search continues until good food sources are identified, including the best ones (Von Frisch, 2014).

2.2. Bees Algorithm

The Bees Algorithm (BA) was proposed by Pham et al. (2006). It is a population based meta-heuristic inspired by the foraging behaviour of bees. BA has both local and global search processes, and it uses a fitness evaluation to perform the search, rather than the probabilistic approach used in many other bee-inspired algorithms. In the literature, two main implementations of the BA exist. The first one is called the Basic BA (BBA), which represents the original BA (Pham, Castellani, and Fahmy, 2008; Yuce et al., 2013). The second main implementation is an improved version of the BBA called Standard Bees Algorithm (SBA), in which new mechanisms are included such as the site abandonment and the neighbourhood shrinking techniques. The earliest proposition of the SBA was introduced in Castellani et al. (2012) and Pham et al. (2012).

Due to its simplicity and its closeness to the real natural behaviour, BA has attracted a significant number of research subjects since its inception. It has been successfully deployed to solve optimization problems in different domains, such as assembly optimization of Printed Circuit Board (PCB) (Pham et al., 2007b; Mei et al., 2010), solving container loading problems (Dereli and Das, 2011), dynamic optimization in chemical engineering (Castellani et al., 2012), solving timetabling problems (Lara et al., 2008; Nguyen et al., 2012; Abdullah and Alzaqebah, 2013), and other applications.

It is worth mentioning that such paradigm has also been adopted in the field of scheduling. In Pham et al. (2007a) a BA was deployed to minimize the total earliness/tardiness penalties on a single machine with common due date. The algorithm gives a promising result compared with discrete particle swarm optimisation (DPSO), tabu search (TS), genetic algorithm (GA), tabu search with genetic algorithm (HTG), and genetic algorithm with tabu search (HGT), and it shows competitive results.

Also, Yuce et al. (2017) proposed a hybrid genetic bees algorithm (GBA) to solve the single machine early/tardy scheduling problem with common due date. In the GBA, researchers have attempted to enhance the BA, by using genetic operators in the global search phase instead of using simple global random search. The results of the GBA were compared with the basic BA and a remarkable improvement of the results was obtained. The main objective of enhancing the global search phase in GBA is to skip the problem of getting stuck in local optima. However, this problem can be caused by the weakness of the local search phase, knowing that the algorithm considers the best solution found to be an elite solution, and that the local search process is performed in the elite solutions. Therefore, escaping from the local optimum can be established by improving the local search process, which is proposed in our approach.

Despite the fact that the BA had successful implementations on several optimization problems, it was noticed that the algorithm is subject to a limited global search capacity, this amplifies the computational complexity of NP-hard optimization issues. Also, the Basic BA runs a higher risk of being trapped in a local optima, which may render the search process totally random (Yuce et al., 2017).

As previously stated, the bees algorithm searches for the best solution to a given optimization problem by mimicking the foraging behaviour of bees. Each point in the search space (potential solution) is considered as a food source. The Basic BA starts by initializing the population to "ns" scout bees, and create "ns" solutions randomly, the cost function

of each solution is evaluated and the solutions are sorted from the best to the worst according to their cost function. The "nb" top ranked solutions are selected to the local search process, and classified into two classes. The first "ne" of the selected solutions belong to the "Elite sites", and the rest "nb-ne" solutions belong to the "non-Elite sites".

The number of recruited bees in each "Elite-site" is set to "nre", and the number of recruited bees in each "non-Elite site" is set to "nrb". The recruited bees perform a local search process in the neighbourhood of each selected solution. The neighbourhood size is set to "ngh". If a recruited bee succeeds in finding a better solution than the previous one, the latter will be replaced by the new solution found. The local search process presents the exploitation phase of the BA.

The global search process consists of performing a random search in the "ns-nb" "non-selected sites", random solutions are then generated and the best solutions replace the worst ones. The global search represents the exploration phase of the BA.

The new population of solutions is sorted according to the cost function value of each solution, and the process is repeated until a sopping criterion is met. Figure 1 represents the flowchart of the basic BA.

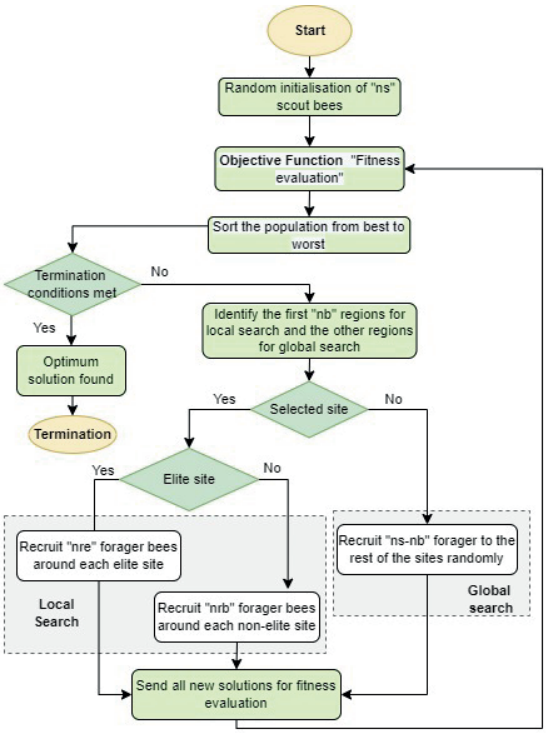

Figure 1. Basic Bees algorithm flowchart.

Table 1. A summary of parameters used in BA.

Symbols	Parameter
ns	Number of scout bees
nb	Number of best selected sites
ne	Number of elite sites with ne < nb
nre	Number of bees in elite sites
nrb	Number of bees in best selected sites
nrns	Number of bees in non-selected sites
ngh	The initial size of patches
Iter	Maximum number of iterations

3. Hybrid BASA for the Single machine scheduling problem

In order to enhance the basic bees algorithm, our proposition consists in improving the process of local search as previously mentioned. The BA moves in the neighbourhood of the solutions with small steps, which offers the ability to exploit an important number of solutions in the neighbourhood. While this approach is excellent for a better exploitation, it causes a problem of being stuck in the local optima. Hence, in order to avoid this frequent shortcoming, we proposed to use the Simulated Annealing (SA) algorithm, which offers the ability to get out of the local optimum by authorizing movements towards solutions of inferior quality compared to the current solution with a certain probability. Such strategy allows a deep exploitation of the elite solutions, with avoidance of getting trapped in local optima.

The principal idea is to improve the ability of the exploitation and exploration in the bees algorithm, by using SA algorithm which helps to intensify the local search process with respect to the global search. Instead of exploiting the elite solutions with simple moves and small steps as in the original BA, the SA can provide better displacements in the neighbourhood of the current solution starting from the best solution previously found.

As known if the algorithm is stuck in a local optimum (ε). It is clear that (ε) is one of the elite solutions found by the algorithm, and the search process will become random because it will be dominated by the global search. Our approach consists in improving the local search for the elite sites which helps in getting out of local optima.

3.1. Solutions representation

As we are dealing with a scheduling problem, the proposed meta-heuristic uses a permutation-based representation, in which solutions are successions of tasks that have to be run on a single machine. A solution then is represented by a vector containing permutations of integers.

3.2. Simulated annealing procedure for the elite sites

The Simulated Annealing Procedure (SAP), presented in Algorithm 1, is one of the proposed strategies in this work, which uses the Simulated Annealing algorithm (Kirkpatrick et al., 1983) to enhance the local search in the elite sites, In the proposed BASA, each one of the bees recruited in the elite site performs a local search by executing SAP. This procedure starts by generating a candidate solution (ω'), from the neighbourhood of the current solution (ω), at each sub-iteration (k_i), i∈{1,2,..., SubIt}, where SubIt is the maximum number of sub-iterations. As known, the SA is based on the metropolis acceptance criterion (Dowsland and Thompson, 2012). The candidate solution is accepted as the current solution based on the acceptance probability denoted by A_k (ω'). Let Δf be the difference between the cost function of the current solution and the cost function of the new solution found, defined by

$$\Delta f = f(\omega') - f(\omega) \tag{2}$$

where $f(\omega)$ and $f(\omega')$ are the cost function values associated with solutions ω and ω', respectively. Hence, the acceptance probability is provided as:

$$A_k(\omega') = \begin{cases} exp\left[\frac{-(\Delta f)}{T_k}\right], & if\ \Delta f > 0 \\ 1, & if\ \Delta f \leq 0 \end{cases} \tag{3}$$

where, T_k is the temperature parameter at iteration k with: $T_k > 0$ for all k and $\lim_{n\to\infty} T_k = 0$

This process will be repeated for a (*SubIt*) number of sub-iterations at each iteration *(k)* with the temperature T_k. At the end of each iteration k, the temperature will be reduced according to equation (4)

$$T_{(k+1)} = \alpha(T_k) \tag{4}$$

where, α is the temperature reduction rate.

The SAP starts from a predefined temperature parameter $T_0 > 0$, and each one of the *(nre)* bees starts the SAP from the same (T_k) value. Each bee executes the SAP for *(SaIt)* number of iterations, and the current solution (ω) will be replaced by the best solution found at the end of the SAP.

Algorithm 1 SAP

```
get the initial solution ω;
T ← T0;
for it = 1:MaxIter do
  for k = 1:SubIt do
    generate new solution from the neighbourhood ω';
    f(ω') ← Objective function (ω');
    Δf ← f(ω') − f(ω);
    if Δf ≤ 0 then
    ω ← ω';
    else
      r ← random();
      if r < exp(−Δf/T) then ▷ metropolis acceptance criterion
        ω ← ω';
      end if
    end if
  end for
    T ← T × α;
end for
```

3.3. Variable Neighbourhood Search like algorithm for the selected non-elite sites

Instead of neighbouring the selected non-elite sites with a single type of movement, it is possible to diversify the movement types in the local search at the selected non-elite sites. This idea is inspired by the Variable Neighbourhood Search (VNS) algorithm (Hansen et al., 2010). Actually, our proposal is to differentiate the neighbourhood exploitation procedure by three types of moves, insertion, reversion, and swap. Each bee performs the roulette wheel selection to opt for one of the aforementioned movements, to move in the neighbourhood. If the resulting solution (ω') is better than the current solution (ω), then (ω) will be replaced by (ω′), and the new solution will be added to the initial population, to compare it with the other solutions see Algorithm 2.

3.3.1. Insertion move

Selecting two random positions from the current solution and insert the first job after the second selected position as shown in Figure 2(a).

3.3.2. Reversion move

The reversion move consists of selecting two positions at random from the current solution, and reverse the order of all jobs in the sequence as shown in Figure 2(b).

3.3.3. Swap move

As it is indicated in Figure 2(c), the swap move consists in selecting two positions at random and perform a pairwise swap between the jobs in the selected positions.

Algorithm 2 VNS-like

get the initial solution ω;

Move ← Roulette wheel to select the movement;

ω' ← Apply Move (ω);

if $f(\omega') > f(\omega)$ then

$\omega \leftarrow \omega'$;

end if

3.4. The BASA Algorithm

The proposed BASA is presented in Figure 3 and the algorithm proceeds as follows. A set of *ns* food sources (potential solutions) is generated randomly, constructing the initial population. After that every visited solution in the initial population is evaluated by calculating its cost function, and the solutions are ranked according to their cost function value from best to worst, in our case the objective function is a minimization function, so the solutions are ordered in ascending order. The candidate solutions are then classified into three categories "Elite sites", "Selected non-elite sites", "Non-selected sites".

For the "Elite sites", *nre* bees are recruited to perform SAP to search better solutions in the neighbourhood of the elite sites, and *nrb* bees are recruited to perform VNS-like algorithm to carry out a local search in the neighbourhood of the Selected non-elite sites, and for the non-selected sites, *nrns* bees are recruited to conduct a global search procedure by generating new random solutions. If a new-found solution is better than the previous solution, this later will be replaced by the new one. This process will be repeated for a given *Iter* number of iterations until a stopping criterion is met.

4. Numerical experiments and discussions

Across these experiments, we tested our approach on a set of benchmarks proposed by Tanaka, Fujikuma, and Araki (2009), which are available on the following site: *(Tanaka's benchmarks,* https://sites.google.com/site/shunjitanaka/sips*)*.

Tanaka's benchmarks have 40, 50, 100, 150, 200, 250 and 300-job problems and each job group contains 125 different problems with a known optimal solutions.

The algorithms were implemented in MATLAB under a PC with Intel Core i5 CPU at 1.0 GHz and 8 Gb RAM in Windows 10 environment.

4.1. Parameters Setting

The parameters setting has an important influence on the computational results. The BASA parameters tuning is performed on two steps: the first one is the

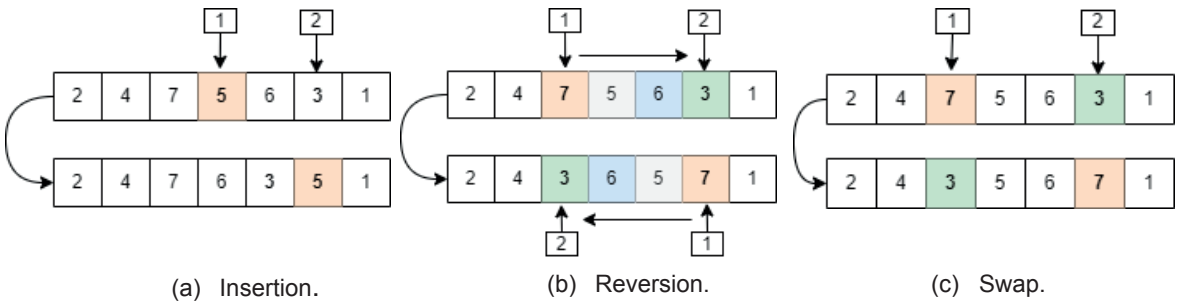

Figure 2. Different movements used in VNS.

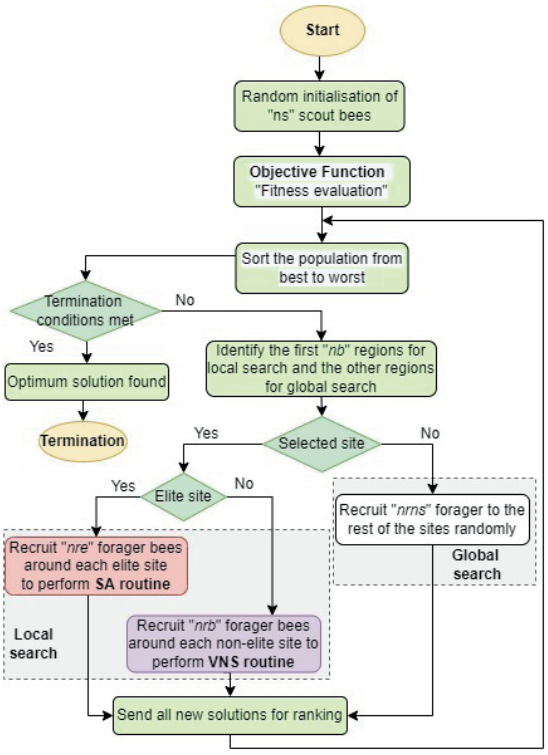

Figure 3. BASA Flowchart.

SA parameters tuning while the second is the BA parameters tuning.

The SA parameters influence the local search process, and they are namely, "*SaIt*" the number of iterations, "*SubIt*" the number of sub-iterations, "*T0*" the initial temperature, and "α" the temperature reduction rate. If the SA goes too deep in the local search, it may lead to a local optimum trapping. Also, if it is going superficially, it will not do the objective behind its utilisation. Thus, the SA parameters are set within the following intervals $SaIt \in [20,40]$, $SubIt \in [30,40]$, $T0 \in [0.01,0.1]$, $\alpha \in [0.70,0.95]$. The empirical experiments show that the best combination of SA parameters [*SaIt, SubIt, T0,* α] is given by values [30, 40, 0.01, 0.90].

The BA parameters affect the overall behaviour of the algorithm, especially the global search process. The BA parameters are, "*iter*" the maximum number of iterations, "*ns*" the number of scout bees, "*nb*" the number of selected sites, "*ne*" the number of elite sites, "*nre*" the number of recruited bees for the elite sites, "*nrb*" the number of recruited bees for selected sites, "*nrns*" the number of recruited bees for the non-selected sites, "*ngh*" the initial size of patches, "*rdamp*" the neighbourhood radius damp rate. BA parameters are set as follows: *iter = 100*, *ns = 20*, *nb = round (0.5´ns)*, *ne = 1*, *nre = 2*, *nrb = round (0.7´ns)*, *nrns = (ns-nb)*, *ngh = 1*, *rdamp = 1*.

The algorithm halts if it reaches the number of iterations *iter=100*, or if the algorithm cannot advance for 30 iterations. It should be mentioned that in the proposed BASA, the shrinking neighbourhood procedure is not used, therefore, the "*ngh*" and "*rdamp*" parameters are set to 1.

4.2. Results and discussions

To verify the performance of the proposed BASA, the following comparison studies are carried out. First, the BASA is compared against the basic BA proposed in (Pham et al., 2006). The objective of these experiments is to elucidate the enhancement of the proposed meta-heuristic. The basic BA original parameters are set as follows:

Iter = 100, *ns = 100*, *nb = round (0.5´ns)*, *ne = round (0.4´nb)*, *nre = 2´nrb*, *nrb = round (0.5´ns)*, *ngh = 1*, *rdamp = 1*.

Second, the proposed BASA is compared against some meta-heuristics extracted from the literature.

4.2.1. Comparing the proposed algorithm results with BA

To evaluate the behaviour of the BASA compared with the BA, the following experiment is set up. A random instance from the benchmark dataset is selected, with tasks number n = 100, and a known exact solution with cost function $f(\omega)$=405122. The algorithms were tested on this instance 10 times and the results are reported in Figure 4.

The convergence curve in Figure 4 showcases that the BASA converges rapidly, and the algorithm is quite stable. In comparison with the BA, the improvement made by the new algorithm is clearly noticeable not only on the quality of the solutions but also on the convergence speed. It can be noticed that the BASA led to a cost function lower than 410000 during 10 iterations whereas the BA needed 100 iterations to reach a cost function of 413500.

To assess the stability and robustness of the BASA, the box-plot of the deviations from the cost function optimal value is illustrated in Figure 5. This figure

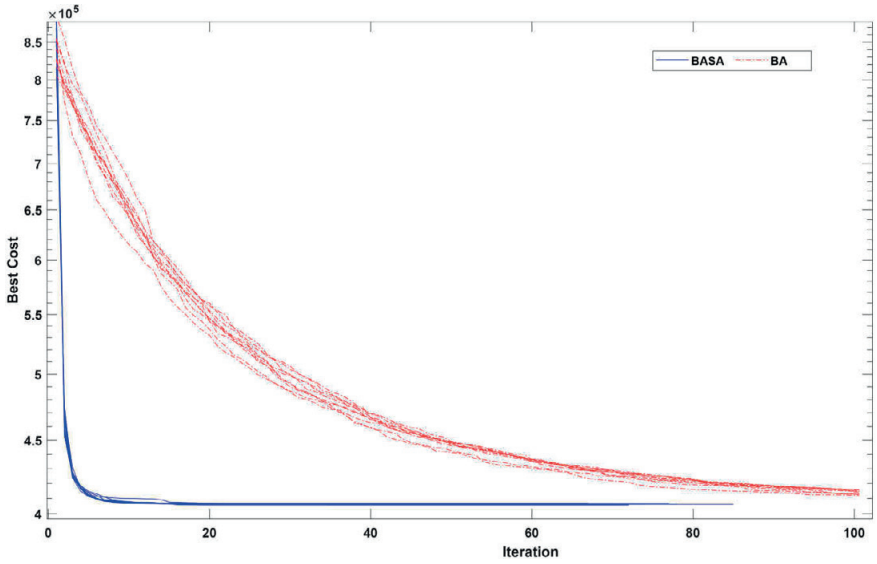

Figure 4. Convergence curves of both BASA and BA.

confirms the robustness of the BASA, the deviations are in the majority grouped nearly around the median value. There are a few individual instances slightly far from the median (represented by + in Figure 5), especially in the instances with *n*=300. This situation can be justified as a consequence of the random behaviour of the algorithm, and is known to be a familiar phenomenon encountered in optimization with meta-heuristics.

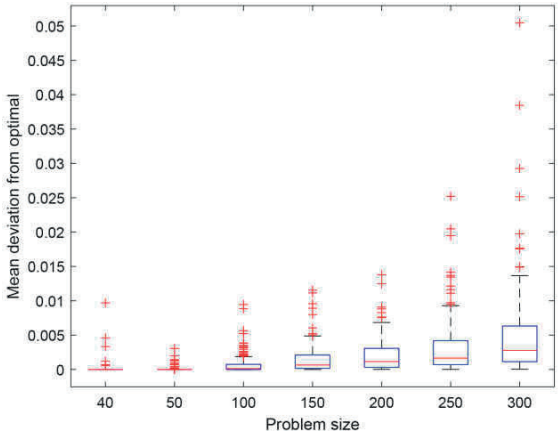

Figure 5 . Box-plot of the deviations from the optimal of all instances.

To evaluate the performance of the proposed BASA, the percentages of mean deviations from optimal (Mean Dev. (%)), and Standard deviations of those deviations (Std. Dev.) are reported. For the first six problem sets (n= 40, 50, 100, 150, 200 and 250) the BASA achieves near-optimal results having less than 0.4% mean deviations. Furthermore, for n=300 problems, BASA reaches solutions with 0.519% mean deviation, which is also acceptable. Moreover, BASA is faster than the basic BA as provided in Table 2. One of the most important well-known advantages of meta-heuristics is their rapid convergence on large instances, which is clearly captured by BASA in comparison to BA requiring a significant amount of time. To determine whether there was a significant difference between the performances of the two algorithms (cost) and the running time (time), a t-test was conducted on the findings for BA and BASA. The goal was to gather evidence to reject the null hypothesis, which assumes no discernible difference between the performances of the two algorithms. A significance level of 0.05 was assigned for the analysis. The "Sign" column in Table 2 displays instances where BASA significantly outperformed the simple BA based on the estimated P-value. The outcomes indicate that there was a statistically significant difference observed in terms of time and cost between the two algorithms across the majority of datasets. However, it should be noted that for datasets with a size of n=40, 50, and 100, the null hypothesis was only rejected for the running time data, while no significant difference was observed in terms of cost".

4.2.2. Comparing the results with other existing approaches

To further validate the effectiveness of the proposed meta-heuristic, the BASA is compared with four

Table 2. BASA results compared with basic BA.

n	Ba			BASA			Sign	
	Mean Dev.%	Std Dev.	Mean Time	Mean Dev.%	Std Dev.	Mean Time	P-value (time)	P-value (cost)
40	0.047	0.001	8.8741	0.016	0.001	2.854	**1,2E-224**	0,499
50	0.422	0.006	11.711	0.012	0.0004	2.942	**2,8E-174**	0,491
100	9.939	0.129	27.328	0.087	0.002	10.268	**2,6E-137**	0,274
150	29.813	0.345	52.513	0.160	0.002	26.451	**7,0E-142**	**0,014**
200	61.398	0.955	80.116	0.209	0.003	46.442	**2,4E-153**	**1,17E-4**
250	84.778	1.156	110.405	0.325	0.004	70.520	**1,0E-131**	**9,7E-08**
300	112.745	1.551	149.353	0.519	0.007	96.571	**1,2E-149**	**6,3E-12**

Table 3. Computational results compared with existing algorithms

	BASA		dABC		ASV		ASS		ASD	
N	Mean Dev. (%)	Std Dev.	Mean Dev. (%).	Std Dev.	Mean Dev. (%).	Std Dev.	Mean Dev. (%).	Std Dev.	Mean Dev. (%).	Std Dev.
40	0.016	0.001	**0.001**	0.010	**0.001**	0.011	2.279	3.479	3.658	5.080
50	0.012	0.0004	**0.005**	0.036	0.006	0.041	1.540	2.018	2.649	3.361
100	0.087	0.002	**0.077**	0.102	0.089	0.146	2.984	4.374	3.890	5.787
150	**0.160**	0.002	0.284	0.129	0.225	0.345	9.043	9.409	11.880	11.960
200	**0.209**	0.003	0.329	0.273	0.298	0.398	16.721	17.893	20.445	20.299
250	**0.325**	0.004	0.334	0.298	0.342	0.519	23.672	25.083	28.188	29.142
300	0.519	0.007	**0.395**	0.401	0.408	0.679	31.564	31.843	37.622	37.700

*The text in **Bold** represents the best Mean Dev. The text Underlined represent the best Std Dev.*

competitive meta-heuristics, which are ASD, ASS and ASV, presented by (M'Hallah and Alhajraf, 2016), and dABC presented by (Yurtkuran and Emel, 2016). These meta-heuristics show excellent performance on Tanaka's benchmark instances and to the best of our knowledge, these are the meta-heuristics which perform the best on the SMSPET.

For simplification reasons, we use the same comparison parameters as in Yurtkuran and Emel (2016). Statistical significance testing was not feasible because the results of competing algorithms were acquired straight from the related research. As a result, the mean and standard deviations are directly compared in this section of the study. Table 3 encompasses the comparison parameters namely, the percentages of mean deviations calculated with respect to optimal value (Mean Dev.) and the standard deviations of these deviations (Std. Dev.)

The results presented in Table 3 reveals that the BASA surpasses all the competitor algorithms in terms of robustness according to the standard deviations (Std Dev.). In terms of quality of solutions, Table 3 confirms that the BASA outperforms ASS and ASD on all benchmark instances, the proposed meta-heuristic statistically outperforms ASV and dABD in problems with a number of jobs $n=150$, $n=200$ and, $n=250$. BASA also has a comparable performance with ASV and dABC in almost problems, except the problems with $n=300$, and the justification of this result for $n=300$ can be clearly understood by looking at the box-plot in Figure 6, where two outliers are noticed so fare from the median of deviations hence affecting the statistical measures. However, it is clearly noticeable that the third quartile in the $n=300$ set, does not exceed *1%* of the mean deviation.

5. Conclusions

In this paper, a novel hybrid Bees Algorithm with Simulated annealing called BASA was presented. The enhancement provided in the BASA was introduced at the level of the local search phase, by performing a Simulated Annealing procedure and Variable Neighbourhood Search like algorithm, instead of the classical search.

The proposed BASA was applied to solve the Single Machine Scheduling Problem including Earliness and Tardiness penalties, with distinct due dates and without considering idle time insertion. Our findings have elucidated an important improvement relative to the basic BA, in terms of running time, quality of solutions, robustness and stability. Conversely, the number of parameters to be tuned in the BASA may be a drawback of the proposed framework and constitutes a challenge for a fine-tune process. Also, more in-depth analysis of the impact of different parameters on the algorithm may be required. The empirical results show that the proposed meta-heuristic can achieve better results against the best meta-heuristics found in literature on the SMSPET by achieving less than 0.6% of the percentage of the Mean deviation from the optimal solution in the worst case. Moreover, the BASA was evaluated to be more robust and stable in comparison with its counterparts.

The results obtained by the BASA motivate future research including BASA in other optimization problems such as job shop scheduling problems. Further, a fine-tuning procedure will also be considered in the objective of reducing the number of parameters in the BASA.

Authors contribution

Ahmed-Adnane Abdessemed: Conceptualization, Methodology, Computations, Data Analysis, Writing - Original draft preparation.

Leila-Hayet Mouss: Supervised the findings of this work, Methodology, Validation, Writing - Reviewing and Editing.

Khaled Benaggoune: Methodology, Data Analysis, Writing - Reviewing and Editing.

References

Abdul-Razaq, T. S., & Potts, C. N. (1988). Dynamic programming state-space relaxation for single-machine scheduling. *Journal of the Operational Research Society*, *39*(2), 141-152. https://doi.org/10.1057/jors.1988.26

Abdullah, S., & Alzaqebah, M. (2013). A hybrid self-adaptive bees algorithm for examination timetabling problems. *Applied Soft Computing*, *13*(8), 3608-3620. https://doi.org/10.1016/j.asoc.2013.04.010

Baker, K. R., & Scudder, G. D. (1990). Sequencing with earliness and tardiness penalties: a review. *Operations research*, *38*(1), 22-36. https://doi.org/10.1287/opre.38.1.22

Castellani, M., Pham, Q. T., & Pham, D. T. (2012). Dynamic optimisation by a modified bees algorithm. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *226*(7), 956-971. https://doi.org/10.1177/0959651812443462

Dereli, T., & Das, G. S. (2011). A hybrid 'bee (s) algorithm' for solving container loading problems. *Applied Soft Computing*, *11*(2), 2854-2862. https://doi.org/10.1016/j.asoc.2010.11.017

Dowsland, K. A., & Thompson, J. (2012). Simulated annealing. *Handbook of natural computing*, 1623-1655. https://doi.org/10.1007/978-3-540-92910-9_49

Hansen, P., Mladenović, N., & Moreno Pérez, J. A. (2010). Variable neighbourhood search: methods and applications. *Annals of Operations Research*, *175*(1), 367-407. https://doi.org/10.1007/s10479-009-0657-6

Ho, Y. C., & Pepyne, D. L. (2001). Simple explanation of the no free lunch theorem of optimization. In *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)* (Vol. 5, pp. 4409-4414). IEEE.

Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, *220*(4598), 671-680. https://doi.org/10.1126/science.220.4598.671

Lara, C., Flores, J. J., & Calderón, F. (2008). Solving a school timetabling problem using a bee algorithm. In *Mexican International Conference on Artificial Intelligence* (pp. 664-674). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-88636-5_63

Lim, C. H., Lim, S., How, B. S., Ng, W. P. Q., Ngan, S. L., Leong, W. D., & Lam, H. L. (2021). A review of industry 4.0 revolution potential in a sustainable and renewable palm oil industry: HAZOP approach. *Renewable and Sustainable Energy Reviews*, *135*, 110223. https://doi.org/10.33932/rir.44.3.2

Mei, C. A., Pham, D. T., Anthony, J. S., & Kok, W. N. (2010, November). PCB assembly optimisation using the Bees Algorithm enhanced with TRIZ operators. In *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society* (pp. 2708-2713). IEEE. https://doi.org/10.1109/IECON.2010.5675114

M'Hallah, R., & Alhajraf, A. (2016). Ant colony systems for the single-machine total weighted earliness tardiness scheduling problem. *Journal of Scheduling*, *19*(2), 191-205. https://doi.org/10.1007/s10951-015-0429-x

Nguyen, K., Nguyen, P., & Tran, N. (2012). A hybrid algorithm of harmony search and bees algorithm for a university course timetabling problem. *International Journal of Computer Science Issues (IJCSI)*, *9*(1), 12.

Pham, D. T., Ghanbarzadeh, A., Koç, E., Otri, S., Rahim, S., & Zaidi, M. (2006). The bees algorithm—a novel tool for complex optimisation problems. In *Intelligent production machines and systems* (pp. 454-459). Elsevier Science Ltd. https://doi.org/10.1177/0959651811422759

Pham, D. T., Koc, E., Lee, J. Y., & Phrueksanant, J. (2007a). Using the bees algorithm to schedule jobs for a machine. In *Proceedings Eighth International Conference on Laser Metrology, CMM and Machine Tool Performance, LAMDAMAP, Euspen, UK, Cardiff* (pp. 430-439).

Pham, D. T., Otri, S., & Darwish, A. H. (2007b). Application of the Bees Algorithm to PCB assembly optimisation. In *Proceedings of the 3rd virtual international conference on intelligent production machines and systems (IPROMS 2007)* (pp. 511-516).

Pham, D. T., Castellani, M., & Fahmy, A. A. (2008). Learning the inverse kinematics of a robot manipulator using the bees algorithm. In *2008 6th IEEE International Conference on Industrial Informatics* (pp. 493-498). IEEE. https://doi.org/10.1109/INDIN.2008.4618151

Pham, Q. T., Pham, D. T., & Castellani, M. (2012). A modified bees algorithm and a statistics-based method for tuning its parameters. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, *226*(3), 287-301. https://doi.org/10.1177/0959651811422759

Seeley, T. D. (2009). *The wisdom of the hive: the social physiology of honey bee colonies*. Harvard University Press. https://doi.org/10.2307/j.ctv1kz4h15

Sourd, F. (2009). New exact algorithms for one-machine earliness-tardiness scheduling. *INFORMS Journal on Computing*, *21*(1), 167-175. https://doi.org/10.1287/ijoc.1080.0287

Sourd, F., & Kedad-Sidhoum, S. (2008). A faster branch-and-bound algorithm for the earliness-tardiness scheduling problem. *Journal of Scheduling*, *11*(1), 49-58. https://doi.org/10.1007/s10951-007-0048-2

Tanaka, S., Fujikuma, S., & Araki, M. (2009). An exact algorithm for single-machine scheduling without machine idle time. *Journal of Scheduling*, *12*(6), 575-593. https://doi.org/10.1007/s10951-008-0093-5

Von Frisch, K. (2014). *Bees: their vision, chemical senses, and language*. Cornell University Press.

Wan, L., & Yuan, J. (2013). Single-machine scheduling to minimize the total earliness and tardiness is strongly NP-hard. *Operations Research Letters*, *41*(4), 363-365. https://doi.org/10.1016/j.orl.2013.04.007

Yau, H., Pan, Y., & Shi, L. (2008). New solution approaches to the general single-machine earliness-tardiness problem. *IEEE Transactions on Automation Science and Engineering*, *5*(2), 349-360. https://doi.org/10.1109/TASE.2007.895219

Yuce, B., Packianather, M. S., Mastrocinque, E., Pham, D. T., & Lambiase, A. (2013). Honey bees inspired optimization method: the bees algorithm. *Insects*, *4*(4), 646-662. https://doi.org/10.3390/insects4040646

Yuce, B., Fruggiero, F., Packianather, M. S., Pham, D. T., Mastrocinque, E., Lambiase, A., & Fera, M. (2017). Hybrid Genetic Bees Algorithm applied to single machine scheduling with earliness and tardiness penalties. *Computers & Industrial Engineering*, *113*, 842-858. https://doi.org/10.1016/j.cie.2017.07.018

Yurtkuran, A., & Emel, E. (2016). A discrete artificial bee colony algorithm for single machine scheduling problems. *International Journal of Production Research*, *54*(22), 6860-6878. https://doi.org/10.1080/00207543.2016.1185550