*engineering proceedings*

MDPI

*Proceeding Paper*

# Using SABC Algorithm for Scheduling Unrelated Parallel Batch Processing Machines Considering Deterioration Effects and Variable Maintenance †

**Ziyang Ji, Jabir Mumtaz * and Ke Ke**

School of Mechanical and Electrical Engineering, Wenzhou University, Wenzhou 325035, China; 13173507056@163.com (Z.J.); kekewzu@163.com (K.K.)

* Correspondence: jabirmumtaz@live.com

† Presented at the 4th International Conference on Advances in Mechanical Engineering (ICAME-24), Islamabad, Pakistan, 8 August 2024.

**Abstract:** This paper investigates the problem of processing jobs on unrelated parallel batch machines, taking into account job arrival times, machine deterioration effects, and variable preventive maintenance (VPM). To address this complex scheduling problem, this paper proposes a Self-Adaptive Artificial Bee Colony (SABC) algorithm, incorporating an adaptive variable neighborhood search mechanism into the algorithm. To verify the effectiveness of the proposed algorithm, we designed comparative experiments, comparing the SABC algorithm with the NSGA-III algorithm on problem instances of different scales. The results indicate that the SABC algorithm outperforms the NSGA-III algorithm in terms of solution quality and diversity, and this advantage becomes more pronounced as the problem scale increases.

**Keywords:** self-adaptive artificial bee colony algorithm; deterioration effects; unrelated parallel batch machine scheduling

check for updates

**Citation:** Ji, Z.; Mumtaz, J.; Ke, K. Using SABC Algorithm for Scheduling Unrelated Parallel Batch Processing Machines Considering Deterioration Effects and Variable Maintenance. *Eng. Proc.* **2024**, *75*, 20. https://doi.org/10.3390/engproc2024075020

Academic Editors: Muhammad Mahabat Khan, Muhammad Irfan, Mohammad Javed Hyder and Manzar Masud

Published: 24 September 2024

**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

Due to machine wear and performance degradation from long-term operation, the processing time of jobs may be affected by machine deterioration effects. Therefore, with the presence of deterioration effects and potential machine interruptions, formulating a scientific and reasonable scheduling plan becomes a key issue for enterprises to improve efficiency and reduce costs.

For the equivalent parallel batch processing machines, Wang and Chou studied the parallel BPM problem of minimizing the makespan. They proposed a hybrid genetic algorithm involving job allocation, sequencing, and batching [1]. Wang et al. studied the hybrid flow shop scheduling problem with practical constraints and proposed an adaptive artificial bee colony algorithm with six search operators that can adjust adaptively to minimize makespan [2]. Li et al. proposed a hybrid artificial bee colony algorithm to address the distributed flow shop batch scheduling problem with deterioration effects. They designed five types of neighborhood structures and a novel scout bee heuristic method to optimize the makespan [3]. Wang et al. studied a distributed assembly line scheduling problem with specific constraints, including factory capacity, transportation capabilities, and setup times. They introduced a Q-learning artificial bee colony algorithm to minimize both makespan and total tardiness simultaneously [4].

Li et al. explored the distributed unrelated parallel machine scheduling problem, considering practical constraints of preventive maintenance. They proposed a partitioned artificial bee colony algorithm. To generate high-quality solutions, the entire population was divided into one employed bee group and three follower bee groups. These four groups employed different search strategies [5]. However, the processing of jobs can cause machine

deterioration, which reduces machine performance and increases job processing time over time [6]. For batch scheduling problems that simultaneously consider deterioration effects and maintenance, Lu et al. studied a class of unrelated parallel machine scheduling problems involving maintenance activities, parallel batch processing, and deterioration effects, aiming to minimize makespan [7].

In the study of preventive maintenance in batch scheduling problems, Beldar et al. focused on investigating issues related to variable preventive maintenance of machines. They targeted the equivalent parallel batch processing machine scheduling problem with deadline requirements and dynamically arriving jobs, aiming to minimize total tardiness. To address this challenge, they proposed two metaheuristic approaches: one based on simulated annealing and variable neighborhood search, and another heuristic method for generating initial feasible solutions. However, in the application of the heuristic algorithm with variable neighborhood search, sorting solutions based on objective values and selecting them for the next generation may potentially diminish the algorithm's exploration efficiency [8].

Compared to other studies, there are fewer investigations using the artificial bee colony algorithm for unrelated parallel batch processing machine scheduling problems that consider deterioration effects and variable maintenance. This paper addresses this issue by proposing an adaptive self-artificial bee colony algorithm and comparing it with the classic Non-dominated Sorting Genetic Algorithm III (NSGA-III).

## 2. Description of the Problem

The problem can be described as follows: processing $n$ diverse jobs on $m$ unrelated parallel batch processing machines, where the processing time is $p_{i,j}$, the job size is $s_j$, and the deadline is $d_j$. The capacity of the batch processing machine $M_i$ is $Q_i$, and its energy consumption rate per unit time is $l_i$. Machines need to consider variable preventive maintenance (VPM) to ensure they are in normal working condition, meaning the continuous processing time or the machine's age cannot exceed a set threshold $UT$. Additionally, based on actual production considerations, job arrival times $r_j$ and machine deterioration constraints are included, meaning the actual processing time of the job changes with the machine's continuous operation time.

The research problem can be represented using the three-field notation as $R_m|p-batch, Q_i, p_{i,j}, d_j, s_j, r_j, VPM|C_{max}, ET, TEC$. The notation is described in detail as follows: $R_m$ represents the machine environment as unrelated parallel batch processing machines. $p-batch$ indicates the batch processing characteristic. $Q_i$ denotes the capacity of machine i. $p_{i,j}$ denotes the processing time of job j on machine i. $d_j$ represents the due date of job j. $s_j$ represents the size of job j. $r_j$ represents the arrival time of job j. VPM denotes variable preventive maintenance. $C_{max}$ denotes the maximum completion time. $ET$ represents energy consumption time. And $TEC$ represents total energy consumption.

## 3. Mathematical Modeling of Deterioration and Preventive Maintenance

For the problem where the deterioration effect of the machine is taken into account, an exponential reliability function is used to describe the reliability $R_{b,i}$ at the $b$ position on the machine $M_i$ ($\lambda$ denotes the equipment breakdown rate), and the reliability $R_{b,i}$ is shown in Equation (1).

$$R_{b,i} = e^{-\lambda \cdot t_{b,i}}, \forall \begin{cases} i = 1, 2, 3, \ldots, m \\ b = 1, 2, 3, \ldots, n \end{cases} \quad (1)$$

Due to the deterioration effects of machines, the actual processing time $dp_{i,j}$ of the job is influenced by machine reliability. Upper and lower reliability thresholds are set as $r_{th1}$ and $r_{th2}$, respectively. When the machine's reliability exceeds $r_{th1}$, the actual processing time of the job is consistent with the base processing time. If the reliability is between $r_{th1}$ and $r_{th2}$, the actual processing time of the job gradually increases as the reliability decreases. If the reliability drops below $r_{th2}$, continuing to process may result in excessively long processing times or machine damage, so the actual processing time is set to infinity. $w$

represents the rate of increase in the job's additional processing time due to the decrease in machine reliability. The piecewise function of the actual processing time of the job with respect to the change in machine reliability is shown in Equation (2).

$$dp_{i,j} = \begin{cases} p_{i,j} & r_{th1} \le R_{b,i} \le 1 \\ p_{i,j} + w \cdot (r_{th1} - R_{b,i}) \cdot p_{i,j} & r_{th2} \le R_{b,i} < r_{th1} \\ \infty & 0 \le R_{b,i} < r_{th2} \end{cases} \quad (2)$$

Preventive maintenance is implemented by setting thresholds for the continuous operating time or usage time of the machines. The threshold UT represents the maximum time a machine can operate continuously before requiring maintenance.

In the SABC algorithm, a maintenance decision improvement strategy is introduced. In each iteration, the current reliability of each machine is checked, and maintenance operations are inserted as needed.

## 4. Self-Adaptive Artificial Bee Colony Algorithm (SABC)

For more complex integration considering deterioration effects and maintenance in unrelated parallel batch processing machine scheduling problems with multi-objective optimization requirements, an adaptive variable neighborhood search mechanism is employed during the maintenance decision improvement phase. A detailed flowchart of the SABC algorithm is shown in Figure 1.

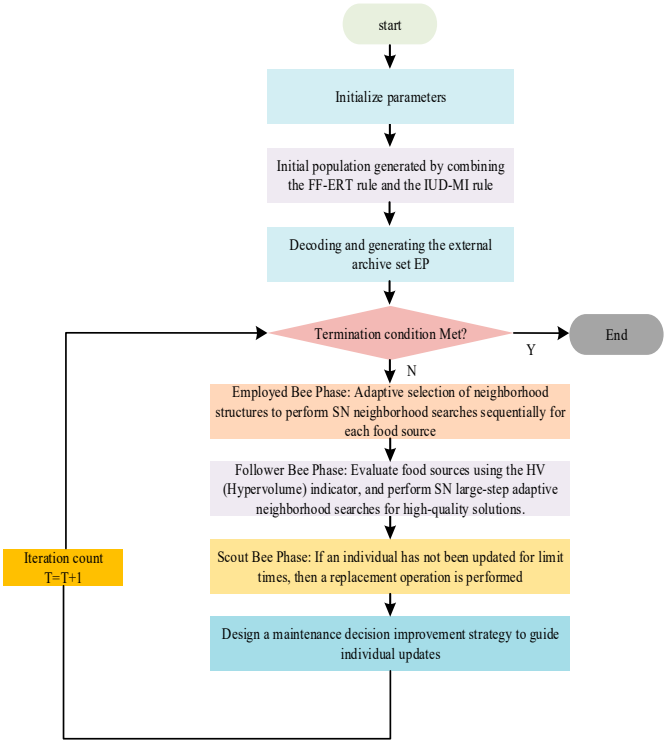



**Figure 1.** Flow chart of the proposed SABC.

We can see that after starting the algorithm, the initial parameters of the Self-Adaptive Artificial Bee Colony (SABC) algorithm need to be set. This includes the population size, number of iterations, and other relevant parameters.

The initial population is generated by combining the FF-ERT and IUD-MI rules. The FF-ERT rule involves randomly assigning jobs to machines and then using the FF rule (First-Fit, where tasks are allocated to the first machine or batch that can accommodate them in the order they arrive) for batching. The batches formed are then sorted using the ERT rule (Earliest Ready Time, where tasks are assigned to the machines or resources that

are ready the earliest). The IUD-MI rule is a more comprehensive heuristic method that batches jobs according to the IUD rule (Insert–Update–Delete, where new tasks are inserted into the existing solution, inappropriate tasks are deleted, and the task sequence is updated to improve the schedule). It then tries different batch sorting rules to increase the chances of finding high-quality initial solutions. Afterward, the MI heuristic rule is used to determine the maintenance positions on the machines, reducing additional processing time.

After generating the initial population, decoding and generating the EP set (External Archive set, used for storing non-dominated solutions) are required. The algorithm checks if the termination condition (program runtime) is met. If the termination condition is met, the algorithm ends. If not, the algorithm proceeds to the employed bee, follower bee, and scout bee phases. Then, a maintenance decision improvement strategy is designed to guide the update of individual solutions, followed by an update of an iteration. At this point, it checks again if the termination condition is met for the next step.

### *4.1. Employed Bee Phase*

Considering the more complex integration of deterioration effects and varying maintenance in the UPBPM (Unrelated Parallel Batch Processing Machine) problem, the Employed Bee Phase adopts an adaptive variable neighborhood search mechanism.

The Employed Bee Phase requires searching near the food source to discover better food sources. The food sources selected by the employed bees undergo SN adaptive neighborhood searches. The neighborhood structures (which involve the exchange of different machine jobs, the insertion of different machine jobs, and the exchange of different machine batches) are as shown in the Figure 2.

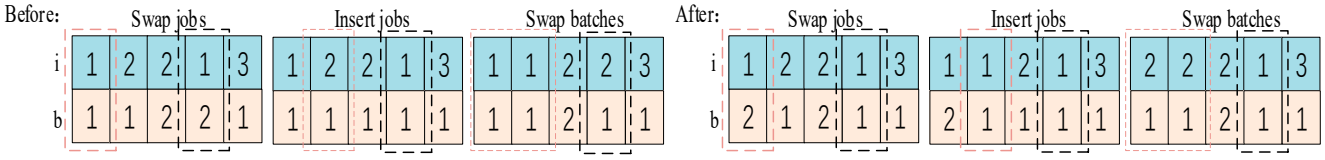



**Figure 2.** Example of neighborhood structure.

### *4.2. Adaptive Variable Neighborhood Search*

In this section, an adaptive variable neighborhood search mechanism is adopted. During each neighborhood search operation, the obtained neighborhood solution may exhibit three different situations, each assigned a different score, namely $\tau_1$, $\tau_2$ and $\tau_3$. Each neighborhood structure $d$ corresponds to a cumulative usage count $u_d$ and a cumulative score $\pi_d$. Each time the neighborhood structure $d$ is used, $u_d$ increases by 1, and $\pi_d$ increases by $\tau_1$, $\tau_2$, or $\tau_3$, respectively. During the learning period LP, the weights of the neighborhood structures are equal. After the learning period LP, the weight $we_d$ takes into account the historical performance of the neighborhood structures $d$ [9], as shown in Equation (3).

$$we_d \leftarrow \begin{cases} 1 & u_d < LP \\ (1-\rho) * we_d + \rho * \frac{\pi_d}{u_d} & u_d \geq LP \end{cases} \quad (3)$$

The selection probability $P_d$ is calculated based on its relative weight $we_d$. The selection probability $P_d$ for each neighborhood structure $d$ is equal to its weight $we_d$ divided by the sum of the weights of all neighborhood structures, as shown in Equation (2). Thus, after the learning period (LP), neighborhood structures with higher weights are more likely to be selected.

This self-adaptive variable neighborhood search mechanism allows the adjustment of the usage probability of various neighborhood structures automatically according to different problem instances and the search process. During the solution retention phase, the following principles are adopted: if the neighborhood solution dominates the original solution, the neighborhood solution is retained; if the original solution dominates the neighborhood solution, the original solution is retained; if the neighborhood solution

and the original solution are in a non-dominant relationship, the solution with the higher individual HV indicator value is chosen.

This selection mechanism, based on dominance relationships and HV indicator values, ensures the continuous improvement of solution quality while preserving the diversity among solutions, helping to avoid the premature convergence of the search process to a local optimum.

$$P_d = \frac{we_d}{\sum_{u=1}^{D} we_u} \tag{4}$$

### *4.3. Follower Bee and Scout Bee Phase*

In the Follower Bee Phase, a neighborhood structure with a larger search step size is designed. In this phase, food sources are selected based on their HV index value using a roulette wheel selection method. Each food source selected by the onlooker bees undergoes SN adaptive neighborhood searches. The main objective of the Scout Bee Phase is to enhance global search. If the objective value of the current food source has not improved, a new individual is generated randomly.

## 5. Examples of Parameters and Results Analysis

### *5.1. Parameters Setting*

The job arrival times are uniformly distributed in the range $[0, 50]$. For small-scale problems, the number of jobs is $n = 10, 20$, and the number of machines is $m = 2$. For medium-scale problems, the number of jobs is $n = 40, 80$, and the number of machines is $m = 3$. For large-scale problems, the number of jobs is $n = 100, 200$, and the number of machines is $m = 5$. For small-scale problems, $\lambda = 0.015$. For medium- and large-scale problems, $\lambda = 0.0075$. The reliability threshold of jobs has an upper limit of $r_{th1} = 0.7$, a lower limit of $r_{th2} = 0.4$, and $w = 2$.

### *5.2. Parameters Tuning*

In this section, based on the results of parameter adjustments, four key parameters influencing the performance of the SABC algorithm were identified using the Taguchi analysis method: these parameters include the population size (N), maximum number of iterations (MAXT), crossover probability ($p_C$), and mutation probability ($p_m$). In small, medium and large scale, the four parameters are shown in Table 1

**Table 1.** The parameters of the algorithms.

| Parameter | SABC | | | NSGA-III | | |
|---|---|---|---|---|---|---|
| | Small | Medium | Large | Small | Medium | Large |
| N | 100 | 100 | 100 | 100 | 100 | 100 |
| MAXT | 40 | 60 | 60 | 110 | 110 | 150 |
| $p_C$ | \ | \ | \ | 0.65 | 0.65 | 0.7 |
| $p_m$ | \ | \ | \ | 0.2 | 0.15 | 0.2 |

### *5.3. Experiments and Discussion*

In this section, the performance of the proposed SABC algorithm is compared with the NSGA-III algorithm. Each algorithm was independently run 10 times on the problem instances, and the average Inverted Generational Distance (IGD), Non-dominance Rate (NR), and Generational Distance (GD) were calculated for both algorithms. Specifically, the experiment evaluates the effectiveness of the algorithms by comparing the performance of SABC and NSGA-III on problem instances of different scales. In each run, the algorithms were executed multiple times independently to ensure the reliability of the results, and the average IGD, NR, and GD for each algorithm were calculated to measure their performance. IGD is used to evaluate the distance between the generated solution set and the reference Pareto front, NR measures the proportion of non-dominated solutions in the solution set,

and GD evaluates the overall quality of the solution set. The experimental data are shown in Table 2.

**Table 2.** The result values of the algorithms.

| Size | n*m* | SABC | | | NSGA-III | | |
|---|---|---|---|---|---|---|---|
| | | IGD | NR | GD | IGD | NR | GD |
| Small | n10m2 | 17.03 | 0.50 | 27.06 | 325.14 | 0.10 | 28.40 |
| | n20m2 | 82.68 | 0.50 | 36.70 | 262.56 | 0.36 | 35.25 |
| Medium | n40m3 | 56.96 | 0.73 | 44.44 | 655.3 | 0.21 | 82.50 |
| | n80m3 | 51.07 | 0.82 | 49.34 | 2761.91 | 0.18 | 119.23 |
| Large | n100m5 | 88.3 | 0.81 | 62.92 | 2134.65 | 0.18 | 171.86 |
| | n200m5 | 236.95 | 0.86 | 68.26 | 6750.03 | 0.14 | 238.75 |

For the small-scale problem with 10 tasks and 2 machines, the SABC's IGD and NR are significantly better than NSGA-III's, indicating that the solutions generated by SABC are closer to the reference Pareto front and contain more non-dominated solutions. Although SABC's GD is slightly worse than NSGA-III's, the difference is not significant. For the small-scale problem with 20 tasks and 2 machines, the SABC's IGD and NR are still better than NSGA-III's, indicating higher quality solutions. However, in terms of the GD metric, SABC is slightly worse than NSGA-III.

For the medium-scale problem with 40 tasks and 3 machines, SABC outperforms NSGA-III in all evaluation metrics. SABC has a smaller IGD, higher NR, and smaller GD, indicating that SABC performs significantly better on this scale. For the medium-scale problem with 80 tasks and 3 machines, SABC significantly outperforms NSGA-III in all metrics. Especially in terms of IGD and NR, SABC's solutions are closer to the reference Pareto front and contain more non-dominated solutions.

For the large-scale problem with 100 tasks and 5 machines, SABC performs excellently in all metrics, far better than NSGA-III. SABC's IGD and NR are significantly better, and GD is much smaller than NSGA-III's. For the largest-scale problem with 200 tasks and 5 machines, SABC outperforms NSGA-III in all evaluation metrics by a large margin. SABC's IGD is much smaller, its NR is significantly higher, and its GD is also much smaller, indicating that SABC has a significant advantage in handling large-scale problems.

## 6. Conclusions

This paper considers job arrival times and machine deterioration effects with variable preventive maintenance constraints, making it more aligned with actual production needs. For this problem, a mathematical model is first constructed, and the SABC algorithm is designed. The SABC algorithm integrates two key modules: the hybrid generation of the initial population using IUD-MI and FF-ERT rules, and the replacement of traditional neighborhood search with a self-adaptive variable neighborhood search mechanism. Through comparison with the NSGA-III algorithm, it is evident that the SABC algorithm significantly outperforms NSGA-III in terms of solution quality and diversity. The conclusion is that the Self-Adaptive Artificial Bee Colony (SABC) algorithm demonstrates superior performance in solving unrelated parallel batch processing machine scheduling problems. Furthermore, the study of UPBPM scheduling problems considering deterioration effects and maintenance provides a foundation for more realistic two-stage hybrid flow shop batch scheduling problems that take into account these factors.

**Author Contributions:** Conceptualization, Z.J.; methodology, K.K. and Z.J.; software, Z.J.; validation, J.M. and Z.J.; formal analysis, Z.J.; investigation, K.K. and Z.J.; resources, K.K.; data—curation, Z.J.; writing—original draft preparation, Z.J., writing—review and editing, J.M. and K.K.; visualization, Z.J.; supervision, J.M. and K.K., project administration, J.M.; funding acquisition: J.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** Basic scientific research project of Wenzhou City (G2023036 & G20240020).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data that support the findings of this study are available from the corresponding author upon reasonable request.

**Acknowledgments:** We would like to express our sincere gratitude to all individuals and organizations that have contributed to this research. We also thank the reviewers for their valuable feedback.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Wang, H.M.; Chou, F.D. Solving the parallel batch-processing machines with different release times, job sizes, and capacity limits by metaheuristics. *Expert Syst. Appl.* **2010**, *37*, 1510–1521. [CrossRef]
2. Wang, J.; Lei, D.; Tang, H. An adaptive artificial bee colony for hybrid flow shop scheduling with batch processing machines in casting process. *Int. J. Prod. Res.* **2023**, *62*, 4793–4808. [CrossRef]
3. Li, J.Q.; Song, M.X.; Wang, L.; Duan, P.Y.; Han, Y.Y.; Sang, H.Y.; Pan, Q.K. Hybrid artificial bee colony algorithm for a parallel batching distributed flow-shop problem with deteriorating jobs. *IEEE Trans. Cybern.* **2019**, *50*, 2425–2439. [CrossRef] [PubMed]
4. Wang, J.; Tang, H.; Lei, D. A Q-learning artificial bee colony for distributed assembly flow shop scheduling with factory eligibility, transportation capacity and setup time. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106230. [CrossRef]
5. Li, Y.; Li, X.; Gao, L.; Zhang, B.; Pan, Q.K.; Tasgetiren, M.F.; Meng, L. A discrete artificial bee colony algorithm for distributed hybrid flowshop scheduling problem with sequence-dependent setup times. *Int. J. Prod. Res.* **2021**, *59*, 3880–3899. [CrossRef]
6. Santos, V.L.A.; Carvalho, T.F.M.; de Assis, L.P.; Weiss-Cohen, M.; Guimarães, F.G. Multi-objective iterated local search based on decomposition for job scheduling problems with machine deterioration effect. *Eng. Appl. Artif. Intell.* **2022**, *112*, 104826. [CrossRef]
7. Lu, S.; Liu, X.; Pei, J.; Thai, M.T.; Pardalos, P.M. A hybrid ABC-TS algorithm for the unrelated parallel-batching machines scheduling problem with deteriorating jobs and maintenance activity. *Appl. Soft Comput.* **2018**, *66*, 168–182. [CrossRef]
8. Beldar, P.; Moghtader, M.; Giret, A.; Ansaripoor, A.H. Non-identical parallel machines batch processing problem with release dates, due dates and variable maintenance activity to minimize total tardiness. *Comput. Ind. Eng.* **2022**, *168*, 108135. [CrossRef]
9. Zhou, S.; Xing, L.; Zheng, X.; Du, N.; Wang, L.; Zhang, Q. A self-adaptive differential evolution algorithm for scheduling a single batch-processing machine with arbitrary job sizes and release times. *IEEE Trans. Cybern.* **2019**, *51*, 1430–1442. [CrossRef] [PubMed]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.