International Journal of Applied Optimization Studies
IJAOS
İSÜ İSTİNYE ÜNİVERSİTESİ İSTANBUL

Efficiently Addressing the Flow Shop Scheduling Problem with Hybrid Obstructions using the Ant Colony Algorithm

MohammadAmin Khedri[1*], Nima Saleh[2], Zahra Sadat Zamani[3]

[1]Department of Industrial Engineering and Futures Studies, Faculty of Engineering, University of Isfahan, Isfahan, Iran

[2]Department of Electrical and Computer Engineering, Faculty of Engineering and Applied Science, University of Toronto, Ontario, Canada

[3]Department of Industrial Engineering, Faculty of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran

*E-mail (corresponding author): ma.khedri@eng.ui.ac.ir

ABSTRACT

The flow shop scheduling problem (FSSP), characterized by the continuous flow of production work across multiple machines, presents formidable computational complexity, necessitating innovative and meta-heuristic approaches for effective solutions. In this research, we harness the potential of the ant colony algorithm (ACA), a promising meta-heuristic technique, to tackle the FSSP incorporating combined obstructions. Our primary objective is to minimize overall production time. This paper introduces various types of obstructions inherent to the FSSP, elucidating the intricacies of the contemporary workshop scheduling challenge. We then present a comprehensive outline of the steps comprising our proposed ACA. To assess the algorithm's effectiveness and practical applicability, we rigorously implement it on a selection of benchmark problems sourced from the existing literature in the field. The results obtained are meticulously compared with optimal solutions, highlighting the algorithm's robust performance. Our findings reveal a promising approach to solving complex FSSP, with notable improvements in efficiency and scheduling optimization. This research represents a significant contribution to the field of flow shop scheduling, offering a viable and innovative solution for addressing intricate production scheduling challenges. The successful application of the ACA in this context underscores its potential for resolving real-world manufacturing dilemmas efficiently.

Paper type: *Research Article*

Received 2022-02-28
Revised 2022-06-15
Accepted 2022-06-19

Keywords:
Scheduling;
Flow Shop;
Combined Obstruction;
Combinatorial Optimization;
Ant Colony Algorithm.

1. Introduction

Within various manufacturing and assembly industries, the execution of tasks necessitates adherence to specific sequences of operations. These sequences often maintain a consistent order, where tasks follow a uniform path. This arrangement of machinery in consecutive order constitutes what is termed a production line. The orchestration of production line scheduling serves as a vital component of operations management, with a multitude of scholarly articles being published in this field annually. In the context of production line scheduling challenges, the primary objective revolves around determining when and which machine should manufacture each product, typically aiming to minimize the total production time.

The inception of production line scheduling as a dedicated field can be attributed to the pioneering work of Johnson (1954). Although Johnson introduced the concept of scheduling, it wasn't until 1965 that the terminology "production line" was explicitly employed (Ignall and Scharge, 1965). A pivotal moment occurred in 1976 when Garey, Johnson, and Sethi (1976) demonstrated the NP-hard nature of production line scheduling, particularly when aiming to minimize makespan in problems involving more than two machines (where $m > 2$). Consequently, solving such complex problems efficiently and effectively hinges on the selection of innovative or metaheuristic algorithms.

In classical scheduling scenarios, the assumption typically holds that once a job is prepared, it can commence immediately on an available machine. Nevertheless, practical situations often introduce constraints related to space, operators, or tools, impeding the immediate initiation of tasks. When a job cannot start on an available machine for any reason, it is termed a "blocking" event. These blocking events can be categorized as follows:

1. RSb (Resource Synchronization Blocking): This occurs when a job delays a machine, preventing it from an immediate start on the subsequent machine.
2. RCb* (Resource Contention Blocking): In this scenario, a job on a machine is held back until the preceding job on the next machine concludes (Trabelsi et al., 2010).
3. RCb (Resource Continuation Blocking): Here, a job on a machine remains idle until the previous job on the next machine not only finishes but also vacates the machine (Martinez de La Piedra, 2005).

In our endeavor to efficiently tackle the FSSP with hybrid obstructions, this study unfolds systematically, culminating in a comprehensive solution. As we embark on our mission to efficiently tackle the FSSP with hybrid obstructions, it is crucial to lay a solid groundwork by navigating through the intricate landscape of prior research and methodologies in Section 2, the Literature Review. Section 3 delves into the heart of the matter by offering a meticulous problem description and mathematical formulation, shedding light on the intricacies inherent in the flow shop scheduling task involving hybrid obstructions. Subsequently, in Section 4, we introduce our chosen Solving Algorithm, the ACA

principles, and explain why it presents a promising solution to this intricate problem. Moving onward, Section 5 is dedicated to presenting the Results and initiating a thorough Discussion. In this section, we assess the performance and efficacy of our proposed approach, drawing comparisons with existing methodologies and highlighting the implications of our findings. Finally, Section 6 concludes our study by summarizing the key takeaways and emphasizing the broader significance of our work in the realm of applied optimization studies.

2. Literature Review

FSSP stands as a classical and foundational challenge in the realm of combinatorial optimization, frequently encountered across a spectrum of manufacturing and production contexts. This intricate problem revolves around the sequencing of a given set of tasks through an array of machines, following a predefined order, with the overarching aim of optimizing specific performance metrics such as minimizing the makespan, total completion time, or overall tardiness. Over time, researchers have dedicated considerable effort to crafting a multitude of algorithms and methodologies designed to grapple with the intricacies of this multifaceted scheduling problem. In this section, we embark on a comprehensive exploration of the pertinent literature concerning the FSSP, the complexities introduced by hybrid obstructions, and the application of the ACA within this domain." The initial exploration into the flow shop scheduling issue can be attributed to Johnson (1954), who introduced a precise methodology for minimizing the makespan in the context of the n-jobs and 2-machines FSSP. As manufacturing companies face intensifying competition marked by increasingly shorter product life cycles, growing demand for custom-made products, and fluctuating demand patterns, the need for efficient scheduling has emerged as an indispensable factor for not only thriving but also surviving in today's fiercely competitive marketplace (Singh & Mahapatra, 2016). The Flow Shop Problem (FSP) is a widely recognized scenario in manufacturing, characterized by a series of tasks or jobs progressing sequentially through multiple production stages. Each stage within this setup is equipped with a single processor or machine (Singh et al., 2016). Pinedo's (2016) research revealed that over a quarter (more than 25%) of manufacturing systems, assembly lines, and information services can be streamlined and conceptualized as flow shop scheduling modules.

González-Neira et al. (2017) delivered a comprehensive survey that offered insights into the realm of flow shop scheduling issues when faced with uncertain conditions. Meanwhile, Cui et al. (2018) introduced an evolutionary, multi-objective approach designed to address robust scheduling challenges in the context of FSSP featuring machine breakdowns and blocking in lot-streaming scenarios. Martinez et al. (2006) contributed significant evidence in 2006, demonstrating that blocking problems incorporating RCb, especially those involving more than four machines, belong to the category of NP-hard problems. The introduction of the RCb* blocking type was attributed to the pioneering work of Trabelsi et al. (2010). Ant Colony Optimization (ACO) is introduced as an innovative metaheuristic

strategy for addressing challenging combinatorial optimization problems in the academic literature, as exemplified by the work of Stützle and Dorigo (2003). The fundamental principle behind the ACO metaheuristic involves the collaboration of a group of artificial ants in the pursuit of high-quality solutions to combinatorial optimization challenges.

One intriguing and significant aspect of ant colonies lies in their foraging behavior, wherein ants exhibit a remarkable ability to discover the shortest routes between food sources and their nest, all without relying on visual cues. During their journeys between food sources and the nest, ants deposit a chemical substance known as pheromone on the ground. Ants possess the capability to detect and respond to these pheromone trails, influencing their path choices probabilistically. Notably, empirical evidence has demonstrated that a collective of ants exhibiting pheromone trail-following behavior can effectively identify the shortest path, as illustrated in the research conducted by Dorigo and Di Caro (1999). Moreover, the introduction of hybrid approaches combining heuristics and metaheuristics, as seen in Ruiz and Stützle (2007), marked an important transition. ACO offers numerous benefits, including notable robustness, effective local optimization capabilities, exceptional global search performance, the potential for parallel computing, and seamless integration with other optimization techniques (Sóbester & Forrester, 2010).

Recently, there has been a surge in the application of metaheuristic algorithms to FSSP. Pioneering work by Dorigo et al. (2006) demonstrated the potential of the ACO algorithm in solving complex FSSP instances. In the realm of the Time-Dependent Vehicle Routing Problem with Time Windows (TDVRPTW), a novel approach emerges. Ye et al. (2018) have introduced a hybrid algorithm that combines the strengths of a Multi-Type Ant System with elements from Ant Colony System and Max-Min Ant System algorithms. Huang et al. (2020) introduced a heuristic approach to tackle the FSSP with sequence-dependent setup times, where startup times were considered sequence-dependent. Comparative assessments against existing algorithms highlighted the efficacy of this heuristic method. Furthermore, Li et al. (2021) extended the application of the honey bee colony algorithm to the FSSP, adapting it for discrete scenarios and applying it to solve problems involving sequence-dependent setup times. The implementation of their algorithm across 780 benchmark instances demonstrated its superiority, achieving approximately a 10% improvement over other algorithms.

According to the reviewed literature, in this research, While ACA has demonstrated its prowess in solving scheduling problems, its application to FSSP instances with hybrid obstructions remains relatively unexplored. Existing research primarily focuses on either traditional FSSP or specific types of hybrid obstructions, failing to provide a comprehensive solution that efficiently addresses the amalgamation of these challenges. Our study seeks to bridge this gap by adapting and enhancing ACA to efficiently tackle the FSSP with Hybrid Obstructions, providing a holistic approach to address the evolving demands of modern manufacturing environments.

In summary, the literature review reveals a significant need for innovative approaches that combine the power of the ACA with the complexity of hybrid obstructions in FSSP. Our research aims to contribute to this emerging area by developing a novel ACA-based solution tailored to this challenging problem domain.

3. Problem Description and Mathematical Formulation

In the context of flow shop scheduling, a set of n jobs ($J = J_1, J_2, \dots, J_n$) needs to be processed on a set of M machines ($M = M_1, M_2, \dots, M_m$). All jobs J_i require a series of identical operations ($O_i = O_{i1}, O_{i2}, \dots, O_{i3}$) that must be performed according to the production process. Operation O_{ij} requires an execution time P_{ij} on machine $M_j \in M$. The objective function used in this article is the longest completion time, denoted as C_{max}.

3.1. Assumptions

In this scenario, the following assumptions were introduced by Baker (1974):

1. Each machine can process only one job at a time.
2. An operation of a job cannot be performed simultaneously on multiple machines.
3. The production process does not experience interruptions during job processing.
4. All jobs are independent and available for processing at time zero.
5. Job setup times on machines are negligible and can be ignored.
6. Machines are continuously available for processing.

4. Solution Algorithm

The algorithm presented in this study draws inspiration from ACO, a heuristic algorithm inspired by the collective foraging of the natural behaviors exhibited by ants in their native environment. ACO has found widespread utility in tackling combinatorial optimization problems, consistently demonstrating its efficiency when compared to alternative algorithms.

The foundation of this algorithm can be traced back to Ahmadizar's (2012) work, building upon the Max-Min Ant System algorithm originally introduced by Stutzle and Hoos (1997). This refined algorithm represents a significant enhancement over the core ACO concept. Notably, within this approach, only the best-performing and contribute to updating pheromone levels a crucial mechanism for maintaining pheromone values within predefined minimum and maximum bounds.

4.1. Initial Pheromone Levels: Slop Algorithm

To establish the initial pheromone levels, the algorithm employs the innovative slope algorithm pioneered by Palmer (1965). The underlying principle of this algorithm is task prioritization based on their processing times. The objective is to minimize the total idle time before commencing the first job on the machines.

The algorithm calculates the slope coefficient for task i using the following equation:

$$A_i = -\sum_{j=1}^{m}((m-(2j-1))p_{ij}), \quad (1)$$

where i denotes the job number, j denotes the machine number, m represents the total number of machines, and P_{ij} represents the execution time of job i on machine j.

The algorithm proceeds as follows:

1. The job with the highest coefficient is selected as the first job to be scheduled.
2. Subsequently, the second job chosen is the one with the highest coefficient among the remaining jobs.
3. This process continues until the last job scheduled is the one with the lowest coefficient.

4.2. ACO Algorithm

The ACO algorithm comprises several essential steps, each contributing to the overall optimization process:

State Transition: In this step, ants construct solutions by iteratively selecting jobs for machines based on pheromone levels and problem-specific heuristics. The probability of selecting a job is influenced by the pheromone level on the associated machine and problem-specific heuristic information.

Local Pheromone Update: Following the construction of each solution, a local pheromone update is performed. This update typically involves a simple evaporation of pheromone levels on all edges (job-machine pairs) within the solution. It prevents excessive pheromone accumulation on the edges.

Global Pheromone Update: After all ants have constructed solutions, a global pheromone update is executed. In this step, pheromone levels are adjusted based on the quality of solutions found by the ants. High-quality solutions receive more pheromones, encouraging other ants to explore similar paths in subsequent iterations.

Local Search: Depending on the specific ACO variant, local search can be an optional step. It entails making small modifications to solutions to explore the neighborhood and potentially enhance their quality. This step contributes to refining the solutions found by the ants.

These steps are iterated for a predetermined number of iterations or until termination conditions are met, such as time limits or convergence criteria. The ACO algorithm leverages the collective intelligence of the ants to search for high-quality solutions in the problem space.

4.3. MAX-MIN Algorithm

Within the ACA, each iteration involves all ants concurrently constructing a solution. Ant k selects its next task using the following rule (Zhang et al., 2014):

$$s = \begin{cases} arg\ max\ (\tau_{ij}) & \text{if } q \leq q_0, \\ s & \text{otherwise,} \end{cases} \quad (2)$$

where q is a random value in the interval [0, 1] and q_0 is a parameter ranging between 0 and 1, and s is a random variable selected according to the probability distribution defined in Eq. (3) (Ahmadizar, 2012).

$$p_{ij}^k = \frac{\tau_{ij}}{\sum_{j \in N_j^k} \tau_{ij}}, i \in N_j^k, \quad (3)$$

where N_j^k is the set of jobs not yet selected, and τ_{ij} represents the relative importance (pheromone level) of placing job i on machine j.

When ant k generates a feasible solution, the associated pheromone levels are updated following the local pheromone update rule (Ahmadizar, 2012):

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \frac{\rho}{C_{max}^k}, \quad (4)$$

where C_{max}^k represents the objective function (longest completion time) for the complete solution generated by ant k and p is a value between 0 and 1.

The local search rule encourages ants to explore a larger space of feasible solutions, contributing to the discovery of more valid solutions within the problem space. Once all ants have generated their solutions, the one with the best objective function (longest completion time) is selected, and its associated pheromones are updated using the global pheromone update rule (Ahmadizar, 2012):

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \frac{\rho z}{C_{max}^{s-best}}, \quad (5)$$

where C_{max}^{s-best} represents the objective function (longest completion time) for the best sequence found in the current iteration, and z is a positive number indicating the relative importance of this sequence. This update ensures that the search continues in the vicinity of the best solution obtained so far. At the end of each iteration, the values of τ_{max} and τ_{min} are updated based on the following formulas (Ahmadizar, 2012):

$$\tau_{max} = \frac{1}{\rho C_{max}^{s-best}}, \quad (6)$$

$$\tau_{min} = U\tau_{max}. \quad (7)$$

These values are then compared to all τ_{ij} values. If τ_{ij} values are greater than τ_{max} or smaller than τ_{min}, they are set equal to the maximum or minimum, respectively.

4.3.1. Initial Pheromone Amount

The initial values of τ_{max} and τ_{min} are determined using the following relationships (Ahmadizar, 2012):

$$\tau_{max} = \frac{1}{\rho C_{max}^{s}}, \quad (8)$$

$$\tau_{min} = U\tau_{max}, \quad (9)$$

where C_{max}^{s} represents the objective function obtained from the Slope heuristic algorithm. Both p and U are parameters that range between 0 and 1. The initial pheromone values are set to τ_{min} initially. The pheromone value update rule for the solution obtained from the Slope method is as follows (Ahmadizar, 2012):

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho/C_{max}^{s}. \quad (10)$$

Eq. (10) increases the pheromone level for sequence S relative to the minimum value, encouraging the search to commence in the vicinity of the initial solution.

4.4. Local Search Algorithm

To enhance the efficiency of the ant algorithm, local search methods can be employed. In this context, the following procedure is utilized:

We evaluate all possible pairs of jobs in the input sequence, and if this evaluation leads to an improved solution, we replace the current sequence with the new one.

4.4.1. Algorithm Steps

1. Initialization of Initial Solution Generation Parameters and Assignment of Initial Pheromones.
2. Repeat the Following Steps Until Termination Conditions Are Met:
   2.1. For Each Ant in the Colony, Repeat the Following Sub-Steps:
      a. Generate a solution by iteratively applying the solution construction rule.
      b. Enhance the solution's quality through local search.
      c. Execute local pheromone updating.
      d. If the solution improves, replace the current best solution.
   2.2. Update pheromone levels according to the final pheromone update rule.
   2.3. Adjust the maximum and minimum pheromone levels to constrain the pheromone values.
3. Present the Best Solution Found.

5. Results and Discussion

The experimentation and subsequent discussion were conducted within a software environment utilizing Delphi, executed on a computer equipped with a Dual Core 3.16 GHz processor and 2GB of RAM.

For the resolution of sample problems through the ACO algorithm, crucial parameters were established as follows: $U = 0.05$, $z = 20$, and $\rho = 0.1$. In addition, the algorithm was deployed with 10 ants, and a predefined maximum of 50 iterations served as the termination criterion. The

parameter q_0 was computed using the formula depicted below, where n corresponds to the total number of jobs in the problem instance:

$$q_0 = \frac{(n-2)}{n}. \quad (11)$$

In the context of result evaluation, the proposed algorithm was rigorously assessed using benchmark data sourced from Trabelsi, Sauvey, & Sauer (2010) . The tabulated findings, indicating the number of problems successfully resolved, are presented in Table 1 for reference.

Table 1. Number of solved examples.

Item	No. of jobs	No. of machines	No. of solved examples
1	5	{5,6,7,10,15,20,50,100}	100
2	6	{5,6,7,10,15,20,50,100}	100
3	7	{5,6,7,10,15,20,50,100}	100
4	8	{5,6,7,10,15,20,50,100}	100
5	9	{5,6,7,10,15,20,50,100}	100
6	10	{5,6,7,10,15,20,50,100}	100
7	11	{5,6,7,10}	100
8	11	{15,20,50,100}	20
9	12	{5,6,7}	100
10	12	{10,15,20,50,100}	20

In this research, a comprehensive comparative analysis was carried out by juxtaposing the obtained results with the optimal solutions documented in a prior publication by the same authors. The examination of the results vis-à-vis the metaheuristic algorithm, as elucidated in (Trabelsi, Sauvey, & Sauer, 2010), yielded the following significant insights:

For problem instances encompassing 5 to 8 jobs, mirroring the scope of our study, it is noteworthy that, in the majority of cases, we were able to attain optimal solutions. Furthermore, the discernible difference in solution times between our approach and the referenced method was generally minimal.

However, as we extended our analysis to encompass larger problem instances, a notable trend emerged. In all instances examined, our algorithm, particularly when implemented with methods "a" and "b," consistently outperformed the comparative method denoted as "c." In the context of the algorithm "c," our approach either yielded superior solutions or achieved solutions that closely approximated its performance. Moreover, an equally noteworthy observation was the substantial reduction in solution times.

Method "a" harnesses a genetic algorithm, employing a population size of 50 chromosomes (individuals).

In contrast, method "b" employs an iterative improvement technique that refines solutions derived from the preceding algorithm, leveraging strategies such as TSS (Total Set Sequence) and NEH (Nawaz-Enscore-Ham). Method "c" encompasses a hybrid approach, amalgamating the best solutions derived from algorithm "b" with a genetic algorithm operating on a population of 100 chromosomes (individuals). This robust comparative analysis underscores the enhanced efficacy of our proposed methods, particularly in tackling larger and more complex problem instances while simultaneously mitigating computational demands.

The examination of results presented in Figures 1 and 2, along with the data elucidated in Tables 2 and 3, offers valuable insights into the performance comparison of various algorithms. Notably, it becomes evident that as the number of jobs (n) increases, the inherent complexity of the problem at hand escalates. In the specific case of n=10, the maximum error observed is approximately 70%. This error margin expands to about 80% when confronted with larger problem instances, such as n=20. This observed trend underscores the direct correlation between problem complexity and the degree of error encountered.

Table 2. Comparison of different solution methods in the case of n=10.

Solution method / Machines	a	b	c	ACO
5	2.31%	5.31%	3.93%	1.69%
10	4.64%	6.01%	5.95%	4.50%
15	15.90%	8.69%	9.77%	7.86%
20	9.57%	8.38%	8.20%	7.56%
50	15.72%	25.48%	9.89%	9.88%
80	44.35%	51.02%	62.69%	44.02%
100	28.84%	23.36%	73.33%	23.11%

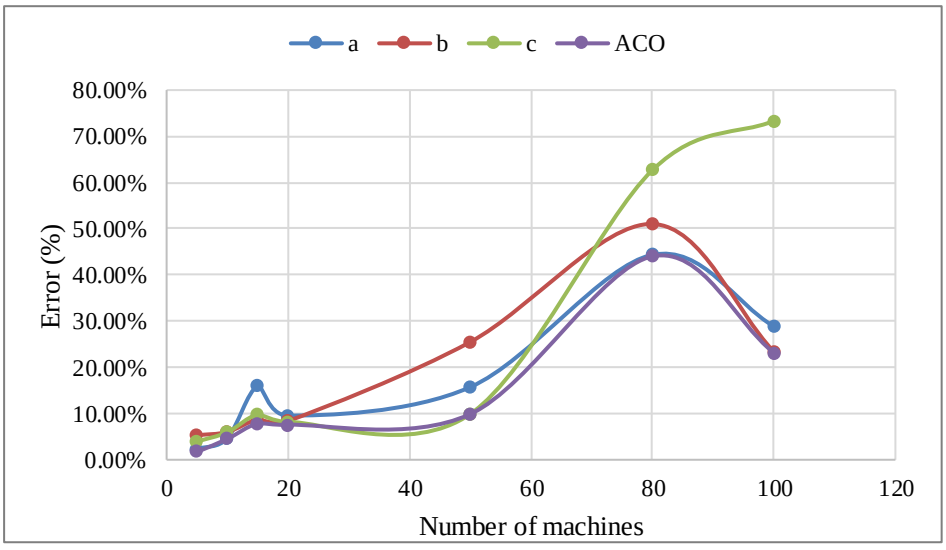


Fig. 1. Error rate for n=10.

Table 3. Comparison of different solution methods in the case of n=20.

Machines \ Solution method	a	b	c	ACO
5	3.12%	1.99%	3.76%	1.82%
10	1.02%	5.21%	4.23%	0.39%
15	5.69%	4.00%	2.39%	1.77%
20	14.36%	2.79%	5.56%	2.57%
50	3.20%	4.61%	41.15%	2.69%
80	9.63%	31.18%	41.27%	9.61%
100	68.69%	83.59%	43.86%	43.72%

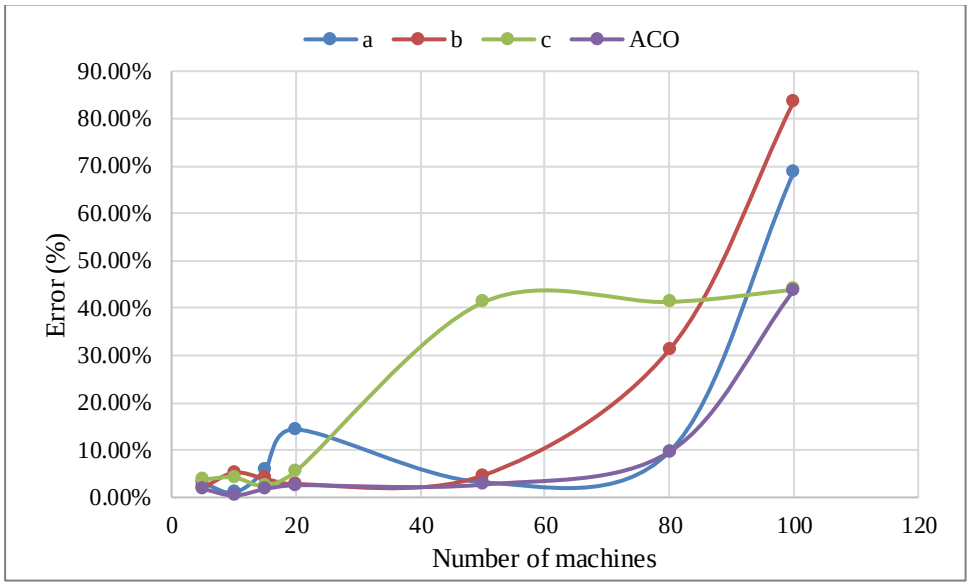


Fig. 2. Error rate for n=20.

Turning our attention to Figure 1, it is readily apparent that, across all scenarios, the error values generated by the ACO algorithm consistently exhibit lower magnitudes when juxtaposed with other competing algorithms. This trend is not a mere coincidence but a robust validation of the ACO algorithm's effectiveness in addressing the problem at hand.

Furthermore, the corroborating evidence in Figure 2 reinforces the prowess of the ACO algorithm. Once again, it emerges as the frontrunner among the compared algorithms, offering sustained superior performance and demonstrating its efficacy in providing solutions with minimized error rates.

6. Conclusion

In conclusion, the meticulous sequencing of job operations on machines undeniably wields a profound influence on production efficiency, punctual fulfillment of customer demand, and the overall cost of manufactured products. This imperative underscores the primary objective within production units, which is to devise operation plans that maximize the utilization of available resources, encompassing

both labor and machinery. Nonetheless, the inevitability of machine breakdowns looms as a persistent challenge, capable of precipitating operational disruptions and amplifying operational expenses.

The job-shop scheduling problem, situated within the domain of combinatorial optimization and production scheduling, stands as a testament to the complexity inherent in manufacturing logistics. Within this challenge, the overarching goal revolves around ascertaining the optimal processing sequence for n jobs across m machines, guided by a predefined objective function, often centered around minimizing the makespan. It is crucial to note that, in this context, the completion of a job hinges on the culmination of all its constituent operations. Traditionally, within the realm of workshop production problems, each job is defined as a collection of indistinguishable items, indivisible even when a subset of items is ready for processing on the subsequent machine. This implies that jobs are treated as independent entities, and the indivisibility of jobs is assumed. However, the real-world manufacturing environment frequently deviates from these idealized constraints.

In the present study, a synergistic amalgamation of the ACO algorithm has been harnessed to enhance solutions for the job-shop scheduling problem, further compounded by the presence of blockages. The robustness of our proposed approach was assessed against benchmark data provided by Mr. Trabelsi and his colleagues, where optimal solutions were already established. The outcomes derived from our algorithm consistently outperformed the results detailed in this article, underscoring its efficacy in addressing this intricate problem domain. Additionally, the temporal aspect cannot be overlooked, as our approach showcased notable variations in solution times, especially in larger problem instances.

Looking ahead, the potential for extending and refining this algorithm to tackle a broader spectrum of job-shop scheduling challenges beckons as a promising avenue for future research. The dynamic interplay between optimization techniques and real-world manufacturing complexities continues to present fertile ground for exploration and innovation.

References

Ahmadizar, F. (2012). A new ant colony algorithm for makespan minimization in permutation flow shops. Computers & Industrial Engineering, 63(2), 355-361.

Baker, K. R. (1974). Introduction to sequencing and scheduling. John Wiley & Sons.

Cui, W., Lu, Z., Li, C., & Han, X. (2018). A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops. Computers & Industrial Engineering, 115, 342-353.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. IEEE computational intelligence magazine, 1(4), 28-39.

Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406),

Dorigo, M., & Stützle, T. (2003). The ant colony optimization metaheuristic: Algorithms, applications, and advances. Handbook of metaheuristics, 250-285.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. Mathematics of operations research, 1(2), 117-129.

González-Neira, E., Montoya-Torres, J., & Barrera, D. (2017). Flow-shop scheduling problem under uncertainties: Review and trends. International Journal of Industrial Engineering Computations, 8(4), 399-426.

Huang, J.-P., Pan, Q.-K., & Gao, L. (2020). An effective iterated greedy method for the distributed permutation flowshop scheduling problem with sequence-dependent setup times. Swarm and Evolutionary Computation, 59, 100742.

Ignall, E., & Schrage, L. (1965). Application of the branch and bound technique to some flow-shop scheduling problems. Operations research, 13(3), 400-412.

Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup times included. Naval Research Logistics Quarterly, 1(1), 61-68. https://doi.org/https://doi.org/10.1002/nav.3800010110

Li, Y., Li, X., Gao, L., Zhang, B., Pan, Q.-K., Tasgetiren, M. F., & Meng, L. (2021). A discrete artificial bee colony algorithm for distributed hybrid flowshop scheduling problem with sequence-dependent setup times. International Journal of Production Research, 59(13), 3880-3899.

Martinez de La Piedra, S. (2005). Ordonnancement de systemes de production avec contraintes de blocage Nantes].

Martinez, S., Dauzère-Pérès, S., Gueret, C., Mati, Y., & Sauer, N. (2006). Complexity of flowshop scheduling problems with a new blocking constraint. European Journal of Operational Research, 169(3), 855-864.

Palmer, D. S. (1965). Sequencing jobs through a multi-stage process in the minimum total time—a quick method of obtaining a near optimum. Journal of the Operational Research Society, 16(1), 101-107.

Pinedo, M. L., & Pinedo, M. L. (2016). Deterministic models: preliminaries. Scheduling: Theory, Algorithms, and Systems, 13-32.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research, 177(3), 2033-2049.

Singh, M. R., & Mahapatra, S. S. (2016). A quantum behaved particle swarm optimization for flexible job shop scheduling. Computers & Industrial Engineering, 93, 36-44.

Singh, M. R., Singh, M., Mahapatra, S., & Jagadev, N. (2016). Particle swarm optimization algorithm embedded with maximum deviation theory for solving multi-objective flexible job shop scheduling problem. The International Journal of Advanced Manufacturing Technology, 85, 2353-2366.

Sóbester, A., & Forrester, A. (2010). J. Ant colony optimization. Alphascript Publishing, 28(3), 1155-1173.

Stutzle, T., & Hoos, H. (1997). MAX-MIN ant system and local search for the traveling salesman problem. Proceedings of 1997 IEEE international conference on evolutionary computation (ICEC'97),

Trabelsi, W., Sauvey, C., & Sauer, N. (2010). Heuristic methods for problems with blocking constraints solving jobshop scheduling. Proceedings of the 8th International Conference on Modeling and Simulation (MOSIM'10),

Ye, D., Wanhong, Z., Hongwei, L., & Yonghui, Z. (2018). Multi-type ant system algorithm for the time dependent vehicle routing problem with time windows. Journal of Systems Engineering and Electronics, 29(3), 625-638.

Zhang, X., Tong, J., & Ma, Y. (2014). An Effective Hybrid Ant Colony Optimization for Permutation Flow-Shop Scheduling. The Open Automation and Control Systems Journal, 6(1).