Applying Genetic Algorithms to Optimize Hybrid Flow Shop Scheduling in Disaster Response Logistics

Zisen Chen [1], Yixuan Guo [2] and Tianyi Ren [3, *]

[1] Haidian Foreign Language Academy, Beijing, 100000, China

[2] Hamden Hall Country Day School, Connecticut, 06053, The United States

[3] Drew University, New Jersey, 07015, The United States

* Corresponding author: tren@drew.edu

Abstract. This paper explains the hybrid flow shop problem and its occurrence in logistics. The paper introduced an example to explain the situation. The example problem is a condition involving different jobs, machines, and stages. The background is in a disaster area where people need supplies. The paper divided the whole process into two stages. The paper applies the genetic algorithm to solve this problem. Each schedule was represented in a form of an individual of the genetic algorithm. The paper narrows the range of the individuals by reintroducing a better individual to improve the fitness value of the individual. The result of the problem when the fitness value was nearly constant. The process of solving the problem is explained in detail. The final schedule was drawn in a graph approximately to scale. The paper explains the flaws of the genetic algorithm. In the code, the values were set to increase the efficiency to get the answer and reduce the flaw. At last, the paper discusses the future use of the hybrid flow shop problem in logistics.

Keywords: Logistics, hybrid flow shop problem, genetic algorithm.

1. Introduction

The logistics industry is closely associated with human life in this fast-paced world. At first, the term logistics was used in war [1]. However, over the last six decades, logistics has become mature enough [1]. Nowadays, the number of users of logistics software is increasing. The efficiency of the logistics process is becoming essential. To improve efficiency, supervisors can apply machine scheduling in the logistics industry. Machine scheduling arranges tasks to optimize productivity, reduce waiting time, and maximize resource utilization [2]. Specifically, machine scheduling involves deciding which machine performs which stage of each task [2]. In a typical parallel machine scheduling problem, n jobs with different processing times are processed on m parallel machines [3].

Machine scheduling can be applied in logistics to improve transportation efficiency. For example, machine scheduling can minimize transportation time and energy consumption [4]. Machine scheduling applications in logistics can optimize the scheduling arrangement of forklifts and automated guided vehicles. With machine scheduling, supervisors can avoid last-minute chaos. Meanwhile, machine scheduling can reduce waiting time, minimize energy consumption, and enhance operational efficiency [4].

In this research, the situation resembled the conditions in a logistics environment. The background was in a disaster area where people needed materials immediately. The objective of the problem was to minimize the time to prepare materials for people in disaster areas. The job was in two stages. The first stage was making the materials. The second stage was packaging the materials. The time to package each material was different by machine. The transportation times of materials were the same because there was no difference in distance between the factory and the disaster area. Therefore, the transportation from the factory to the disaster area was not a stage in this research. Machines can manufacture and package all materials in the problem. This situation is a typical hybrid flow shop scheduling (HFSP). Each job follows the same route of multiple stages in the HFSP [5]. Multiple parallel machines process the jobs in each stage [5]. The solution to the problem is the optimal schedule that follows the rule of an objective function [5].

The research used the genetic algorithm to solve the HFSP problem. The genetic algorithm is a technique introduced by Goldberg and Holland based on principles of natural selection [6]. The genetic algorithm involves procedures that solve complex multi-objective optimization problems by constantly improving generations [6]. The genetic algorithm can minimize processing time, maximize resource utilization, and balance workloads. The genetic algorithm begins with a population of individuals [7]. In this research, each individual represented a schedule that can finish the problem. The genetic algorithm evaluates the individuals' fitness value and allocates reproductive opportunities [7]. The algorithm improves the individuals' fitness value by selection [7]. In the research, the individuals' crossover and mutation created new individuals. The individuals with the better fitness were more likely to pass into the next generation, and the generation would generate and select individuals repeatedly until the final generation. The research ensured the fitness value would be nearly constant in the final generation. In this way, the individuals in the last generation would be close to the fittest individuals. The research upgraded the answer by introducing the first individual to get the final individual. The final individual had the highest fitness value for the research. The upgrading continued until no individual had a minimum completion time less than that of the final individual. Therefore, the final individual had a higher probability of giving the ideal schedule for the problem. The final individual also gave the final schedule for the problem. However, there was more than one final schedule because two schedules had the same completion time.

2. Methodology

2.1. Problem Description

The research applied the genetic algorithm to solve the HFSP. According to a research, genetic algorithm is effective in solving scheduling problem [8]. The processing of materials involved two stages. The two stages were manufacturing the materials and packaging materials. In two stages, there were n materials and M machines. Each material required processing at the two stages. Every machine is available for processing each stage for a material. Table 1 shows a case with 4 materials and 3 machines.

Table 1. Processing Time

Materials	Stages	m0	m1	m2
J0	s00	11	14	16
	s01	4	2	4
J1	s10	12	14	13
	s11	6	6	3
J2	s20	11	7	6
	s21	3	5	5
J3	s30	7	9	12
	s31	4	3	1

The meanings of the notations are defined: N represents number of materials, M_i represents index of machines, $i \in \{0, \dots, M\}$, J_j represents index of materials, $j \in \{0, \dots, N\}$, S_{jk} represents the process where the stage k of material j is processed, T_{jk} represents the time to finish stage k of material I, C_{max} represents the completion time of the process.

The objective function is $minC_{max}$. The function refers to the minimization of the completion time. The objective function follows constraints. Each stage cannot be interrupted after the stage starts. Only one machine is responsible for a single stage of processing a material. There is no prioritization of any material. A machine cannot process more than one material at the same time.

2.2. Genetic Algorithm Process

2.2.1. Initialize population

The initial population was composed of individuals generated [9]. The population size should always be less than the number of possible schedules through the research. Therefore, the initial population size was 200.

2.2.2. Interpretation of individuals

The problem ensured each stage was assigned to an appropriate machine. Therefore, the individuals were represented in the form of eight (job, stage, machine). A sample individual for the problem is shown: [(0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 2), (3, 1, 2)]. With the interpretation, the research could find out the detailed schedule management.

2.2.3. Evaluate fitness

The research applied the objective function to calculate the fitness of each individual. The fitness value evaluates the individual's quality in solving the problem [9]. In this research, the completion time was the only determinant of work quality. Individuals with higher fitness values had a shorter time to complete the processing procedure. For example, a schedule with a completion time of 19 hours had a higher fitness value than a schedule with a completion time of 22 hours.

2.2.4. Selection

The selection ensured better individuals passed on to the next generation and eliminated the ineffective individuals. The selection was based on the roulette wheel method in this research. The basic theory of the roulette wheel model is randomly select a number between 0 and 1 [10]. The first individual with a cumulative probability that is larger than the selected number would survive [10]. Since the fittest individuals were more likely to be selected, they would be more likely to survive to the next generation. In this way, the selection eliminated the worst individuals by comparing the fitness value.

2.2.5. Reproduction

New individuals were created for the next generation by the crossover and mutation. Crossover exchanged part of the two chromosomes and generated new individuals. For example, when there were two individuals [(0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 2), (3, 1, 2)] and [(0, 0, 2), (0, 1, 0), (1, 0, 2), (1, 1, 1), (2, 0, 3), (2, 1, 1), (3, 0, 2), (3, 1, 1)], and the second node of the individuals were crossover, the new individuals would be [(0, 0, 0), (0, 1, 0), (1, 0, 1), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 2), (3, 1, 2)] and [(0, 0, 2), (0, 1, 1), (1, 0, 2), (1, 1, 1), (2, 0, 3), (2, 1, 1), (3, 0, 2), (3, 1, 1)]. The crossover rate was 0.75 in the code because the algorithm made sure new individuals could be generated and avoided omitting an effective individual.

For mutation, parts of the individual's chromosome were varied. For example, an individual [(0, 0, 0), (0, 1, 0), (1, 0, 1), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 2), (3, 1, 2)] could be mutated into a new individual [(0, 0, 2), (0, 1, 0), (1, 0, 1), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 2), (3, 1, 2)] by mutating the first node. The mutation rate was 0.15 in the code to make sure that mutation would occur but will less likely generate ineffective individuals.

The reproduction ensured new individuals can be created during the process and the setting of the crossover rate and mutation rate ensured the individuals were more likely to be effective.

2.2.6. Generating the solution

The process of evaluate fitness to reproduction repeated in each generation. The research set the greatest number of generations to 500 to ensure time efficiency of the calculation. The population size of the later generations is 50. In this way, the code will give the answer efficiently. The research chose the individual with the highest fitness value at the five-hundredth generation. The code might not be effective. Therefore, the research repeated the answer to find a new schedule. The research found the difference in completion time between the two schedules. In this way, the effectiveness of the code could be ensured. In this research, the algorithm could be shown as effective if the difference between the two schedule was less than 7. This is showing the algorithm is narrowing the range of individuals.

The whole process of generating the solution was done again by introducing the first solution. When introducing the first solution, the population size was 100. the crossover rate was 0.65, and the mutation rate was 0.1. If the answer changed after introducing the first solution, then the research would continue by introducing the new individual. When repeatedly introducing a better individual, the individual occurred next will be closer to the best schedule. When the individual did not change or no better individual was found, the process would stop. The research set the individual occurred at last as the best schedule since it did not change. At last, the schedule was drawn in a graph to show the schedule is effective.

3. Results and Discussion

3.1. The Initial Schedule

The first schedule given by the code is [(0, 0, 0), (0, 1, 1), (1, 0, 2), (1, 1, 2), (2, 0, 1), (2, 1, 1), (3, 0, 0), (3, 1, 2)]. The minimum time to complete processing that is given by the solution is 19 hours. Fig. 1 draws the schedule to visualize the order of each process. The graph shows the sequence of sjk done on machine i. The graph is approximately drawn to scale. There is a gap between s11 and s31 because the first stage of material 3 is done before the second job starts. Since the schedule is finished until the last job has finished, the job management before the s31 done is relatively free.

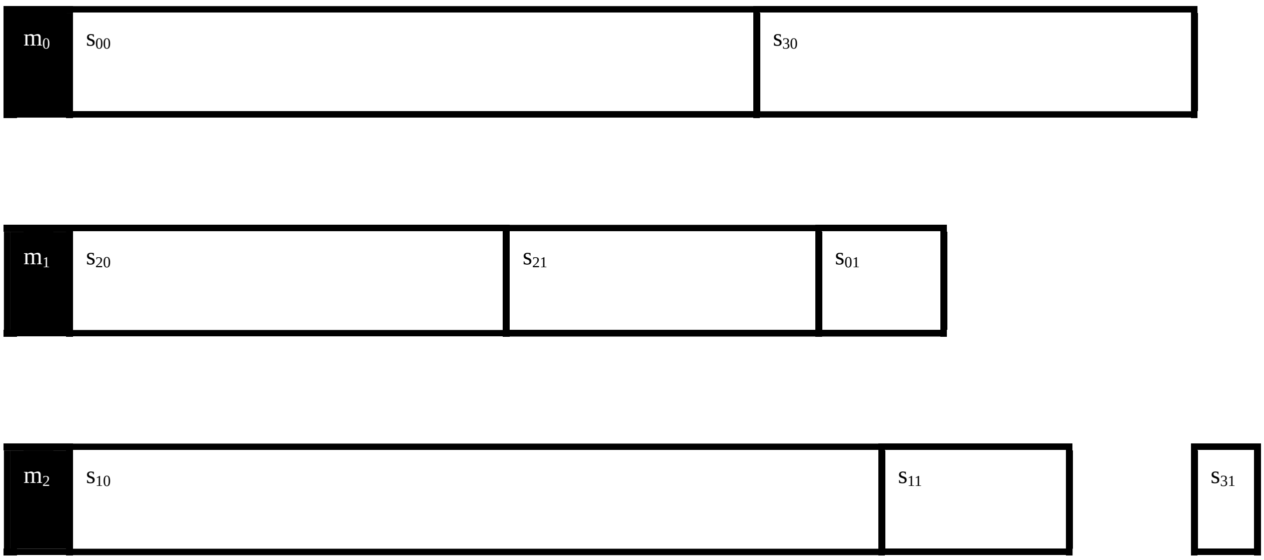

Figure 1. The Initial Schedule

The minimum completion time given by the schedule is $t_{00} + t_{30} + t_{31} = 19$. The individual given by the program is one of the schedules that can be complete during a program. The solution represents that the minimum time the factory can finish processing the materials is nearly 19 hours. However, the answer may not be exactly 19 hours. Therefore, the research did the process again using the same code again to find out the completion time different from the previous solution. The scheduling method given by the program at this time is [(0, 0, 1), (0, 1, 1), (1, 0, 2), (1, 1, 2), (2, 0, 0), (2, 1, 2), (3, 0, 0), (3, 1, 0)], and the finished time is 22. The minimum processing time given by the solution has a difference of 3 hours with 19 hours. In order to show the schedule can give the exact minimum time, the schedule is drawn in Fig. 2.

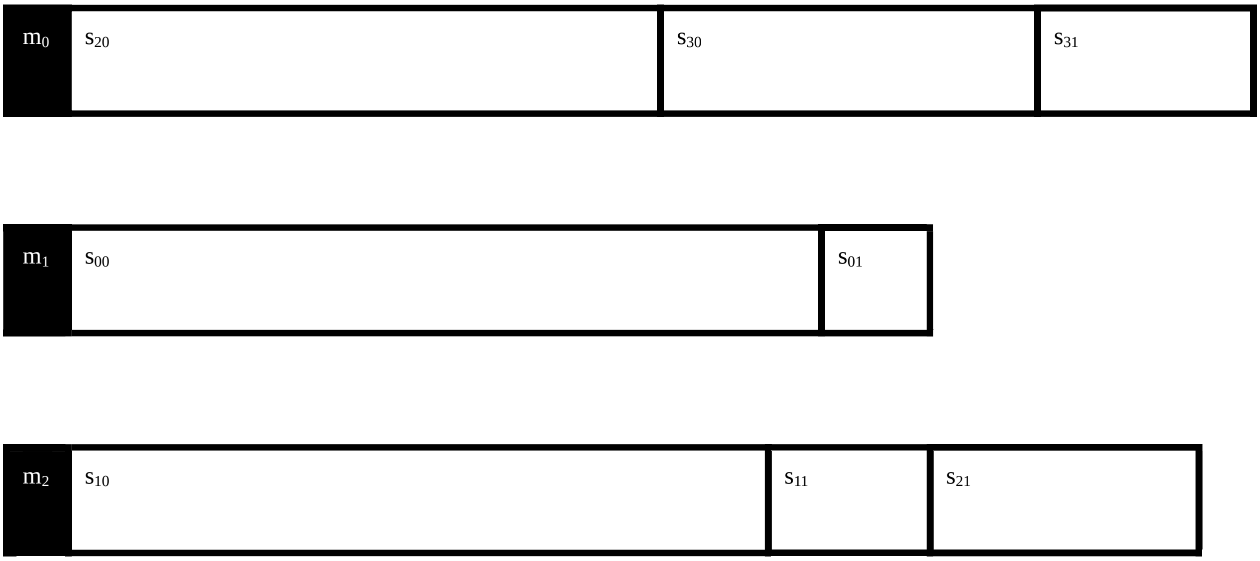

Figure 2. The Second Schedule

The minimum completion time given by the schedule is $t_{20} + t_{30} + t_{31} = 22$. Although the completion time was different, some nodes of the individual remain the same. Therefore, it shows that the genetic algorithm is narrowing the range of completion time. The research did ten more repetitions. In the repetition, the same answer occurred again. There were also schedules that generated the same finish time to 19 hours. However, no schedule gave a minimum completion time of less than 19 hours. All the schedules in the repetition gave a completion time lie between 19 hours and 22 hours. This is a sign that the range of the solution should be less than 22 hours. However, the first code was not effective enough to generate the final answer. Therefore, a step of upgrading the answer is needed.

3.2. Final Schedules

The first schedule was introduced into the previous process. The solution changed to [(0, 0, 0), (0, 1, 1), (1, 0, 2), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 1), (3, 1, 2)]. The given minimum time to process the materials was 18 hours this time. The research continued to upgrade answer by introducing better individuals. In ten times of introducing better individual, some individual appeared again. However, all individuals generate a completion time of 18 hours or larger. More importantly, the schedule that gives the completion time of 18 hours was the same as the schedule the research introduced. Therefore, the individual may be the best individual. In this way, Fig. 3 draws the schedule to make sure the individual is effective.

The Fig 3 shows the schedule is effective, and the machine that did the last job is machine 1. Therefore, the finish time is $t_{20} + t_{30} + t_{01} = 18$. However, there is not only one schedule that has the finish time of 18 hours.

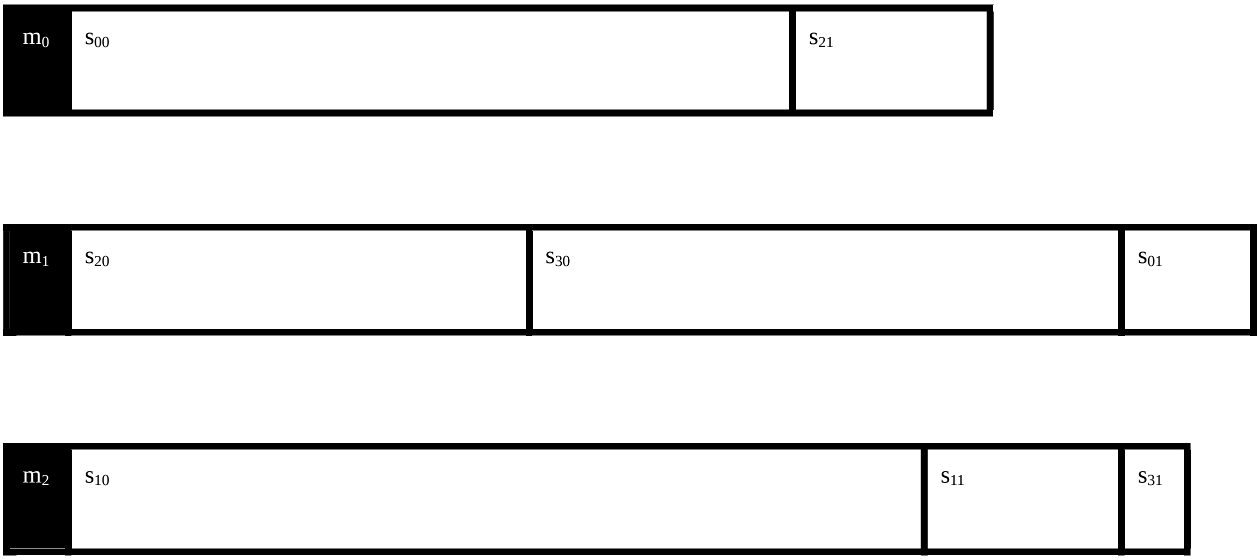

Figure 3. The Final Schedule (1)

The research introduced [(0, 0, 1), (0, 1, 1), (1, 0, 2), (1, 1, 2), (2, 0, 0), (2, 1, 2), (3, 0, 0), (3, 1, 0)] into the process. The answer changed to [(0, 0, 0), (0, 1, 0), (1, 0, 2), (1, 1, 2), (2, 0, 1), (2, 1, 0), (3, 0, 1), (3, 1, 2)]. The completion time is still 18 hours. In this way, the research's hypothesis that there is only one best schedule can be refused. In the Fig. 4, the schedule is drawn to visualize the solution. However, compare with the third schedule the final schedule did not change significantly. It is a mutated individual of the third schedule.

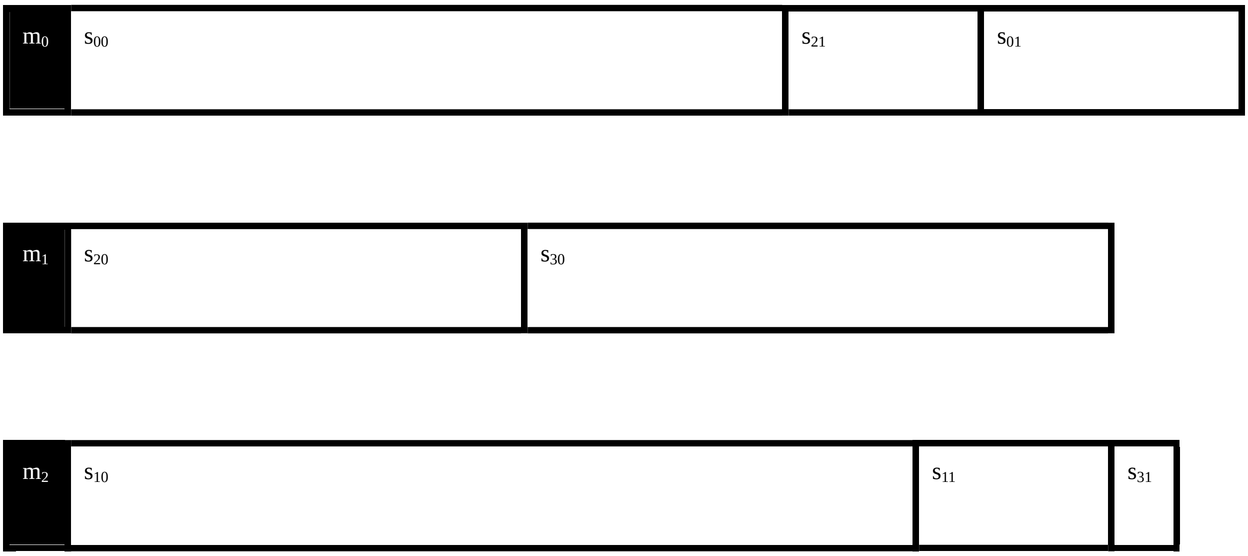

Figure 4. The Final Schedule (2)

The schedule is effective and the completion time is $t_{00} + t_{21} + t_{01} = 18$. However, there is no confident evidence to prove the last two schedules are the best because the genetic algorithm does not involve testing the effectiveness of the answer. Since there was no better schedule generated by the code, 18 hours is the closest to the answer to the problem but may not be the best schedule.

Although the essay used the genetic algorithm to solve the problem, the genetic algorithm still has few flaws. For example, among the individuals generated by the first code, the answer lies in an area between 19 and 22. Although the range is small in this problem, the range might increase if the problem is more complex. The population size is 200 in this problem, but it is not a constant value in all of the problems. The value would increase in other complex problems. More importantly, the genetic algorithm requires large calculation to finish the whole process. Therefore, it is not effective in all kinds of problems. In order to solve more complex questions, the genetic algorithm must be improved.

4. Conclusion

This research applied machine scheduling on a logistics condition. The preparation process before sending materials is the situation of the problem. The program shows that the result of the problem is not single. The schedule that is close to the best schedule is generated at last by introducing previous individuals. This research solved a HFSP in a logistics environment. The final schedule is a example of how schedule is interpreted and visualized. More importantly, machine scheduling can be an effective method whenever supervisors need to minimize the time to complete the transportation of goods. However, the problem in the research was not complicated and would only happen out of normal logistics conditions. This research narrowed the logistics process to making and packaging materials. Transporting goods is also one of the stages in a logistics condition. More importantly, most machines cannot process two distinct stages of work in a logistics condition. Therefore, the problem design is under an ideal condition. Future experimenters should add more stages and vary the machines between stages to solve the problem. Although the problem may become complex, there will be better methods to solve the application of machine scheduling for the problem if the genetic algorithm is developed and improved.

Authors Contribution

All the authors contributed equally and their names were listed in alphabetical order.

References

[1] Neil Southern R. Historical Perspective of the Logistics and Supply Chain Management Discipline. Transportation Journal, 2011, 50 (1): 53 - 64.

[2] Jan Karel Lenstra, David B Shmoys, Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. Mathematical Programming, 1990, 46: 259 - 271.

[3] Wang Zhenbo, Xing Wenxun Parallel machine scheduling with special jobs. Tsinghua Science and Technology, 2006, 11 (1): 107 - 110.

[4] Shaoxing Wu, Li Liu. Green Hybrid Flow Shop Scheduling Problem Considering Sequence Dependent Setup Times and Transportation Times. IEEE Access, 2023, 11: 39726 - 39737.

[5] Victor Fernandez-Viagas. A speed-up procedure for the hybrid flow shop scheduling problem. Expert Syst. Appl, 2022, 187: 115903.

[6] David E. Goldberg, John H. Holland. Genetic algorithms and machine learning. Machine Learning, 1988, 3 (23): 95 - 99.

[7] Sourabh Katoch, Sumit Singh Chauhan, Vijay Kumar. A review on genetic algorithm: past, present, and future. Multimed Tools Appl, 2021, 80: 8091 – 8126.

[8] Eva Vallada, Rubén Ruiz. A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. European Journal of Operation Research, 2011, 211 (3): 612 - 622.

[9] Ceyda Oĝuz, M. Fikret Ercan. A genetic algorithm for hybrid flow-shop scheduling with multiprocessor tasks. Journal of Scheduling, 2005, 8: 323 - 351.

[10] Adam Lipowski, Dorota Lipowska. Roulette-wheel selection via stochastic acceptance. Physica A: Statistical Mechanics and its Applications, 2012, 391 (6): 2193 - 2196.