A Genetic Algorithm for Scheduling Laboratory Rooms: A Case Study

Rafael Fuenmayor[1], Martín Larrea[1], Mario Moncayo[1], Esteban Moya[1], Sebastián Trujillo[1], Juan-Diego Terneus[1], Robinson Guachi[1], Diego H. Peluffo-Ordoñez[2,3,4], and Lorena Guachi-Guachi[1,3(✉)]

[1] Department of Mechatronics, Universidad Internacional Del Ecuador, Av. Simon Bolivar, 170411 Quito, Ecuador
{gufuenmayorso,jalarreafr,mamoncayolo,esmoyata,jutrujillopr,juterneusgo, roguachigu,loguachigu}@uide.edu.ec

[2] Modeling, Simulation and Data Analysis (MSDA) Research Program, Mohammed VI Polytechnic University, Lot 660, Hay Moulay Rachid Ben Guerir, 43150 Ben Guerir, Morocco
peluffo.diego@um6p.ma

[3] Smart Data Analysis Systems Group (SDAS Research Group), Ben Guerir 47963, Morocco
{peluffo.diego,lorena.guachi}@sdas-group.com

[4] Faculty of Engineering, Corporación Universitaria Autónoma de Nariño, Pasto 520001, Colombia
diego.peluffo@aunar.edu.co
https://sdas-group.com/

Abstract. Genetic algorithms (GAs) are a great tool for solving optimization problems. Their characteristics and different components based on the principles of biological evolution make these algorithms very robust and efficient in this type of problem. Many research works have presented dedicated solutions to schedule or resource optimization problems in different areas and project types; most of them have adopted GA implementation to find an individual that represents the best solution. Under this conception, in this work, we present a GA with a controlled mutation operator aiming at maintaining a trade-off between diversity and survival of the best individuals of each generation. This modification is supported by an improvement in terms of convergence time, efficiency of the results and the fulfillment of the constraints (of 29%, 14.98% and 23.33% respectively, compared with state-of-the-art GA with a single random mutation operator) to solve the problem of schedule optimization in the use of three laboratory rooms of the Mechatronics Engineering Career of the International University of Ecuador.

Keywords: Genetic algorithms · Mutation · Scheduling optimization

1 Introduction

The use of laboratories is critical in professional training because practical sessions supplement theoretical foundations and ensure that concepts are clarified

H. Florez and H. Gomez (Eds.): ICAI 2022, CCIS 1643, pp. 3–14, 2022.
https://doi.org/10.1007/978-3-031-19647-8_1

and emphasized appropriately. Problems frequently arise in the planning of class schedules and the use of laboratories in various educational facilities. Crossing schedules for students or professors, as well as excessive time for handling constraints, prevent the various initial courses of a career from having access to laboratory facilities, giving priority only to those courses at the end of the careers. These difficulties are typically caused by the laboratory's capacity, the availability of teachers, and the hours during which the laboratory can be used.

Recent research for scheduling problems have introduced the genetic algorithms (GAs) and a combination between GAs and reinforcement learning, as the most widely used and effective techniques for optimizing resources such as time and space availability. For instance, works in [4–8,10,12,13,16–19,21,22] introduce GAs which develop a series of processes from a population of candidate solutions to a specific problem to obtain the best possible solutions. On the other hand, works in [2,14,28], use reinforcement learning algorithms to obtain a solution from the definition of different parameters that represent the starting point of the genetic algorithms. Although there are several works on schedule optimization, each one focuses on a specific problem, and its application in the optimization of schedules for the use of laboratories in the Mechatronics career of the International University of Ecuador demands an adjustment and optimization for the specific conditions of the problem to be solved.

Therefore, in this work, we present the implementation of a GA with a controlled mutation to produce an optimal schedule for the use of laboratories at the International University of Ecuador's Mechatronics Engineering career. The main differences regarding [21], which optimize class scheduling, lie in the control strategy used and the application to the optimization of laboratory practice use.

This paper is structured as follows: Sect. 2 presents a brief overview of the state-of-the-art. The description of the database used for the experimentation is presented, together with the definition of parameters and characteristic operators of the GA in Sect. 3. Then, in Sect. 4, the experimentation process is described together with the metrics used to evaluate the performance of the algorithm. Finally, the obtained results and conclusions are presented in Sects. 5 and 6 which supports that the modification presented in the mutation operator improves the convergence time, efficiency of the results and the fulfillment of the constraints of 41%, 36.36% and 25% respectively, in comparison with [21].

2 Related Works

Evolutionary algorithms are techniques based on various theories of biological evolution in which only the strongest and best adapted individuals within an environment will be able to survive and reproduce; that is, a competition is generated from an initial population of individuals coexisting in a specific environment with limited resources, resulting in a selection of the individuals best adapted to the environment. Through mutation and recombination processes,

these individuals become the progenitors of a new generation; these new individuals will now compete among themselves, resulting in an increase in the population's quality.

Taking advantage of these characteristics, several works have used evolutionary algorithms to find an efficient solution for schedule generation problems in various academic and industrial areas. Typically, the generation of schedules or project planning requires the coexistence of multiple factors and resources, which greatly complicates such problems [3–11,13,18,20–23,25,27,29].

For instance, in [13], a genetic algorithm with specific constraints to maximize the fitness value of each individual is proposed, which after several generations arrives at the solution of a research oriented scheduling problem. Hybrid evolutionary algorithms also provide even more optimized solutions. In this regard, [29] offer good results in solving resource-constrained project scheduling problems (RCPSPs) by using the sequential performance of multi-operator algorithms under two sub-populations, which provides a modified forward and backward justification approach to obtain feasible schedules.

Recently, evolutionary algorithms have been used to solve schedule optimization problems is the optimization of resources and labor time in the various involved tasks; for example, the hybridization of artificial intelligence (AI) with GA increases the search capacity for a near-optimal solution while avoiding most premature convergence problems [3,5–8,10,11,13,21,23].Other applications include maximizing the utilization of operating rooms, minimizing the cost of operating time overruns and minimizing the cost of waste due to unused time. The elitist search technique is the most commonly used to optimize this kind of problems, which in minor surgery problems outperforms other applications previously tested. Whilst in complex surgeries, this technique produces satisfactory results [3,6,16].

Although evolutionary algorithms have been widely used for scheduling optimization problems, their performance is dependent on different parameter combinations, so they must be redesigned for any extension of the problem [6]. There are cases where the redesigned algorithm may lose diversity and reach a point where it may not converge; as a result, certain stopping conditions, such as a maximum number of generations, population diversity, control of the fitness function, and so on, can be implemented [20].

In order to reduce time-consuming parameter setting tasks, some works have adopted reinforcement learning algorithms, in which a decision is made on various parameters that will provide support to reach an optimal solution to a specific problem [1,2,14,15,26,28]. In [24] a technique of waiting time priority genetic algorithms (WTPA) is also implemented for the reduction of the total average time of all the activities of a schedule; obtaining good results and achieving the proposed objective.

3 Materials and Methods

Inspired by [21], a GA for engineering course scheduling, we propose a GA to solve a scheduling generation problem to optimize the use of the laboratories, with the goal of reducing convergence time and restriction compliance.

To solve our problem of laboratory scheduling in the career of Mechatronics in the International University of Ecuador, we used the number of courses, the number of rooms of the laboratory, the number of subjects (electronics, mechanics, ...), and the number of students per subjects as shown in Table 1.

Table 1. Data description of the laboratories of the mechatronics career

Data	Description	Value
Labs	Available labs	3 Labs
Available times	Times in which the labs are available	13 times
Professors	Professors than can impart the subjects	8 Professors
Subjects	Subjects that need to use the laboratories	17 subjects
Students per subject	Number of students that take a certain subject.	Electronics1: 17, Mechanics3: 10, etc.
Lab capacity	Max number of students per lab	Lab1: 25, Lab2: 10, Lab3: 15

The flowchart of the proposed GA is depicted in Fig. 1. It involves three major steps: Initialization, selection & combination, and mutation based on [21].

Initialization of the Population: It creates the first generation of the population which is going to be used for the solution. The first generation is generated randomly using certain parameters, such as which subjects belong to which academic level and which professors can teach which subject. Following that, the fitness score of the population is calculated by taking into account the number of conflicts found on each generated schedule.

Selection: Once the fitness value for each individual in the population has been calculated using the selection operator, the individuals with the highest fitness value are chosen to create the next generation. It allows the algorithm to get closer to the optimal schedule or solution with each generation. Individuals in this application are chosen using a tournament technique to find the fittest individuals. In this work, we set the number of selected individuals to 3 (Tournament selection size = 3).

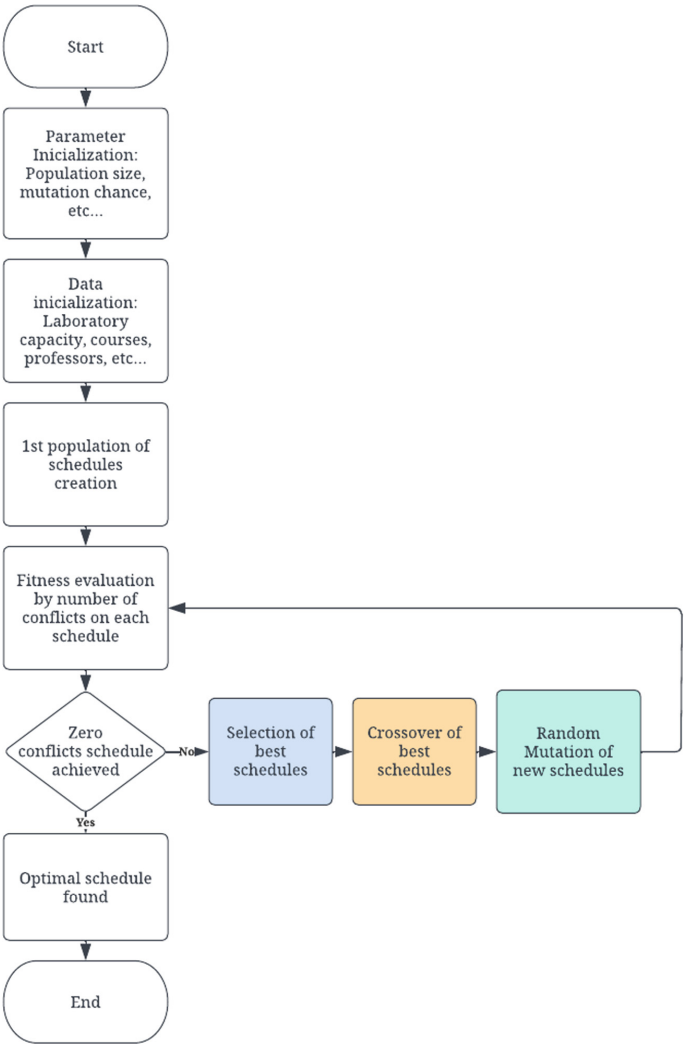

Fig. 1. The genetic algorithm's workflow for generating an optimal schedule for the use of laboratory rooms.

Crossover: The crossover operator uses the parents chosen from the previous generation to create a new one by combining the genetic information of the fittest parents, with the goal of producing populations with higher fitness values. In this work, we use a random number to determine how much genetic information will be taken from each parent involved, randomly mixing the characteristics, and thus creating completely new individuals for the next generation.

Mutation: This is the last operator in the process of implementing the genetic algorithm. Consist in modifying the genetic information of the newly created generation. In this work, a mutation rate of 0.1 is set (mutation rate = 0.1), which determines the probability of an individual being mutated; this value is chosen to ensure the variety of individuals while maintaining its quality. If a random number is less than the mutation rate, the schedule or individual is mutated or generated again. The mutation is used to maintain diversity among individuals in a population, avoiding problems such as the GA not converging quickly enough.

Once these operators are used, a fitness evaluation of the population is performed again. If there is a schedule with no conflicts, the algorithm stops and presents this as the optimal solution. Otherwise, the process is repeated until a solution is found.

Proposed Mutation Operator (Controlled Mutation). In order to improve the efficiency of the results, in terms of convergence, due to increase diversity and respecting the fulfillment of the constraints regarding the algorithm presented in [21], a controlled mutation operator is proposed as depicted in Fig. 2:

In [21], the mutation process was carried out using a single mutation value that was compared with a random number. If the randomly generated number is less than this value, a change is made in the vector that stores the courses, with one of them being replaced by a new course.

In contrast, the proposed controlled mutation operator employs two mutation values (0.05; 0.1), resulting in the generation of two conditionals. In the first one, if the random number is less than 0.05 then one of the first ten courses will be chosen at random and replaced by a new course. In the second one, if the random number is between the mutation rates, a course between position 13 and 17 (of the vector in which the courses are stored) will be randomly chosen to be exchanged with a newly generated course.

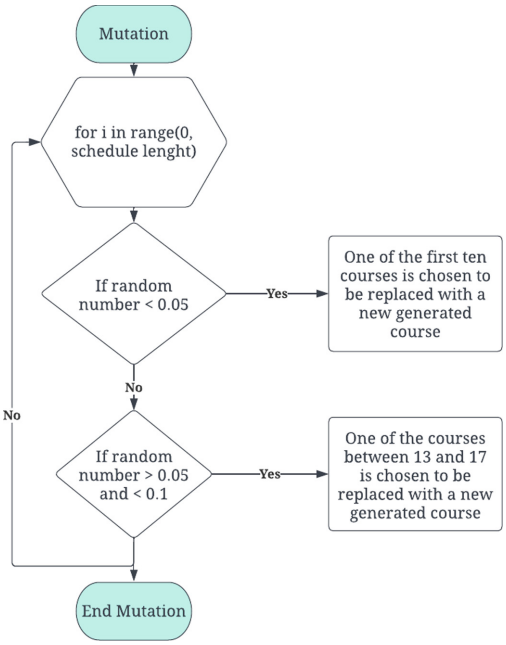

Fig. 2. A controlled mutation operator.

4 Experimental Setup

For the implementation of this algorithm, the Google Colaboratory programming environment with the Python language was used; the environment has 100 GB of storage space and 12 GB of RAM. The Random library was used to maintain randomness in different operators such as mutation and crossover. Prettytable was used to create an appropriate visualization of the results (it provides the possibility of designing a tabular representation of the resulting schedule).

The ability of the proposed GA algorithm with a controlled mutation operator for the use of laboratory rooms was evaluated in terms of the convergence time, effectiveness, and compliance of restrictions.

- **Convergence time:** this metric measures the time required by the algorithm to find an optimal solution free of conflicts.
- **Result effectiveness:** it evaluates the fitness of the individuals of the last generation. While the best individual will always have a fitness value of 1 (there is no conflict), the efficiency is determined by the fitness of the worst individual of the last generation (if the fitness number is lower, then the efficiency will be worse).
- **Compliance of restrictions:** This metric corresponds to the number of conflicts presented by the last generation's individuals. This metric is summarized in the number of conflicts experienced by the worst individual of that generation since the best individual has 0 conflicts.

For comparison purposes, each algorithm was executed 10 times and the execution times, number of generations and the established metrics were registered. The parameters used in the implementation of the GA are summarized in Table 2.

Table 2. Parameter values.

Parameter	Description	Value	Value selection procedure
Population size	Parameterized the number of individuals within the population	9	This value was determined experimentally.
Selection pressure	Parameter that allows to set how many individuals (schedules) are going to be selected to be parents	1	Parameters specified in base algorithm [18].
Mutation rate	Parameter compared with a random number which determines if a mutation is done or not	0.1 and 0.05	This value was determined experimentally

Python routines of the GA are available at:
https://github.com/mamoncayolo/LabSchedulingGeneticAlgorithm.git

5 Experimental Results

From Table 3, it can be observed the variation of the execution time for the proposed GA and the base GA with single random mutation [21]. The average execution time through ten independent executions exhibits that the proposed GA is faster than the base algorithm with a minimum value of 1.469s and a maximum value of 2.03s. On the other hand we have the base algorithm with a minimum time of 1.696s and a maximum time of 3.754s.

Table 3. Execution time required by the proposed GA and the state-of-the-art.

	Execution time (seconds [s.])												
	Execution number										MIN	MAX	AVG
	1	2	3	4	5	6	7	8	9	10			
GA with controlled mutation	1.647	1.48	1.93	1.52	1.775	2.03	1.965	1.469	1.639	1.757	1.469	2.03	1.721±0.21
GA with single random mutation [21]	1.696	2.46	2.27	2.314	2.851	3.055	3.754	1.754	1.605	2.514	1.696	3.754	2.427±0.67

From the Fig. 3, it can be seen that the proposed GA algorithm exhibits a minimum value of 0.2 (fitness) and a maximum value of 1, being this the maximum value of an optimal schedule. On the other hand the base algorithm

exhibits a minimum value of 0.167 and a maximum value of 1 as well. However the proposed GA algorithm achieves the highest average value of 0,327 ± 0,0716, against 0,278 ± 0,0756 reached by the base GA algorithm.

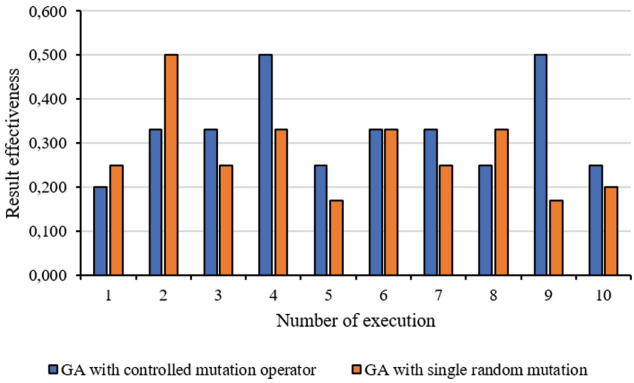

Fig. 3. Result effectiveness of the proposed GA with controlled mutation operator vs GA with single random mutation.

For the constraint compliance metric, the number of conflicts found in the schedules or individuals is taken into account. For this reason, as shown in the Fig. 4, the higher the value, the worse the result. In the proposed GA algorithm there is a minimum value of 1 (number of conflicts) and a maximum value of 4. On the other hand we have the base algorithm with a minimum value of 1 and a maximum value of 5.

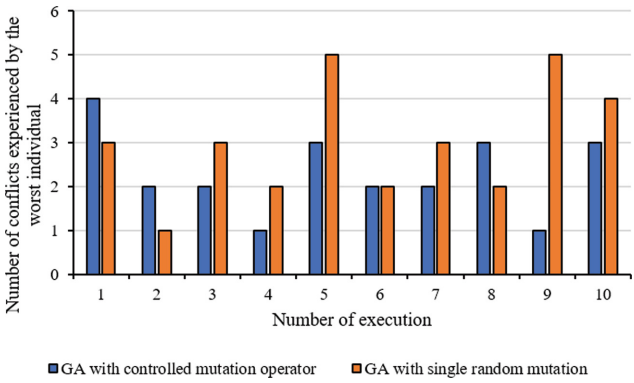

Fig. 4. Compliance of restrictions of the proposed GA vs the GA with a single random mutation.

According to Fig. 5, even though the base algorithm obtains a minimum value of generations less than the proposed GA, this does not imply that the base

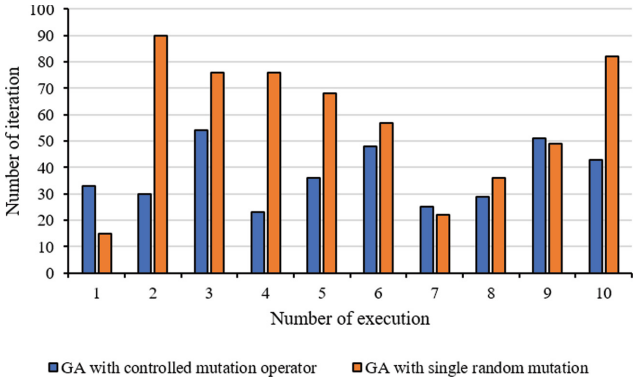

Fig. 5. Number of iterations of the proposed GA vs the GA with a single random mutation.

has better results, because even if it took fewer generations to find an optimal schedule, it took more time to find it.

6 Conclusion

This work presented a genetic algorithm capable of solving the problem of generating an optimized schedule for the use of three laboratory rooms of the Faculty of Mechatronics Engineering. After the experimentation comparing the proposed algorithm with the base algorithm [21], a remarkable improvement in all the analyzed metrics was evidenced. An average improvement of approximately 29% in the convergence time, 14.98% in the efficiency of the results and 23.33% in the fulfillment of the constraints was obtained. Based on these results, it can be concluded that the modification in the mutation operator of the algorithm maintains a balance between diversity and survival of the fittest individuals, which is reflected in the improvements in the metrics. A disadvantage of this application is the fact that, if one wishes to increase the size of the data set, one should reconsider the parameter values and certain conditions in the fitness function.

For future work, a user interface could be generated in which data can be entered according to the user's needs, in order to have a more versatile program. Likewise, it is recommended to experiment with different hybrid operators that can improve the performance of the genetic algorithm.

Acknowledgments. This work is supported by SDAS Research Group (https://sdas-group.com).

References

1. Alhuniti, O., Ghnemat, R., El-Seoud, M.S.A.: Smart university scheduling using genetic algorithms. In: Proceedings of the 2020 9th International Conference on Software and Information Engineering (ICSIE), pp. 235–239 (2020)

2. Alomari, K., Almarashdi, O., Marashdh, A., Zaqaibeh, B.: A new optimization on harmony search algorithm for exam timetabling system. J. Inf. Knowl. Manage. **19**(01), 2040009 (2020)
3. Amindoust, A., Asadpour, M., Shirmohammadi, S.: A hybrid genetic algorithm for nurse scheduling problem considering the fatigue factor. J. Healthcare Eng. **2021** (2021)
4. Amjad, M., Butt, S., Anjum, N., Chaudhry, I., Faping, Z., Khan, M.: A layered genetic algorithm with iterative diversification for optimization of flexible job shop scheduling problems. Adv. Prod. Eng. Manage. **15**(4), 377–389 (2020)
5. Ansari, R., Saubari, N.: Application of genetic algorithm concept on course scheduling. In: IOP Conference series: Materials Science and Engineering, vol. 821, p. 012043. IOP Publishing (2020)
6. Asadujjaman, M., Rahman, H.F., Chakrabortty, R.K., Ryan, M.J.: An immune genetic algorithm for solving NPV-based resource constrained project scheduling problem. IEEE Access **9**, 26177–26195 (2021)
7. Chen, R., Yang, B., Li, S., Wang, S.: A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. Comput. Ind. Eng. **149**, 106778 (2020)
8. Chen, X., Yue, X.G., Li, R., Zhumadillayeva, A., Liu, R.: Design and application of an improved genetic algorithm to a class scheduling system. Int. J. Emerg. Technol. Learn. (iJET) **16**(1), 44–59 (2021)
9. Doğan, A., Yurtsal, A.: Developing a decision support system for exam scheduling problem using genetic algorithm. Eskişehir Tech. Univ. J. Sci. Technol. A-Appl. Sci. Eng. **22**(3), 274–289 (2021)
10. Donoriyanto, D.S., Silfiana, I.Y., Pudji, W.E., Suryadi, A., Widodo, L.U.: Determination of maintenance schedule of loading and unloading pump machine using genetic algorithm method. J. Phys. Conf. Ser. **1569**, 032008 (2020)
11. Ha, V.P., Dao, T.K., Pham, N.Y., Le, M.H.: A variable-length chromosome genetic algorithm for time-based sensor network schedule optimization. Sensors **21**(12), 3990 (2021)
12. Herrera-Granda, I.D., Martín-Barreiro, C., Herrera-Granda, E.P., Fernández, Y., Peluffo-Ordoñez, D.H.: Forthcoming paper icor2020-90b35-01 a hybrid genetic algorithm for optimizing urban distribution of auto-parts by a vertex routing problem
13. Idroes, R., Maulana, A., Noviandy, T., Suhendra, R., Sasmita, N., Lala, A., et al.: A genetic algorithm to determine research consultation schedules in campus environment. In: IOP Conference Series: Materials Science and Engineering, vol. 796, p. 012033. IOP Publishing (2020)
14. Kakkar, M.K., Singla, J., Garg, N., Gupta, G., Srivastava, P., Kumar, A.: Class schedule generation using evolutionary algorithms. In: Journal of Physics: Conference Series, vol. 1950, p. 012067. IOP Publishing (2021)
15. Köksal Ahmed, E., Li, Z., Veeravalli, B., Ren, S.: Reinforcement learning-enabled genetic algorithm for school bus scheduling. J. Intell. Transp. Syst. **26**(3), 269–283 (2022)
16. Li, X., Chen, H.: Physical therapy scheduling of inpatients based on improved genetic algorithm. In: Journal of Physics: Conference Series, vol. 1848, p. 012009. IOP Publishing (2021)
17. Lin, Y.-K., Chou, Y.-Y.: A hybrid genetic algorithm for operating room scheduling. Health Care Manage. Sci. **23**(2), 249–263 (2019). https://doi.org/10.1007/s10729-019-09481-5

18. Liu, J., Liu, Y., Shi, Y., Li, J.: Solving resource-constrained project scheduling problem via genetic algorithm. J. Comput. Civil Eng. **34**(2), 04019055 (2020)
19. Lorente-Leyva, L.L., et al.: Optimization of the master production scheduling in a textile industry using genetic algorithm. In: Pérez García, H., Sánchez González, L., Castejón Limas, M., Quintián Pardo, H., Corchado Rodríguez, E. (eds.) HAIS 2019. LNCS (LNAI), vol. 11734, pp. 674–685. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29859-3_57
20. Mammi, H.K., Ying, L.Y.: Timetable scheduling system using genetic algorithm for school of computing (tsuGA). Int. J. Innov. Comput. **11**(2), 67–72 (2021)
21. Nugroho, A.K., Permadi, I., Yasifa, A.R., et al.: Optimizing course scheduling faculty of engineering unsoed using genetic algorithms. JITK (Jurnal Ilmu Pengetahuan dan Teknologi Komputer) **7**(2), 91–98 (2022)
22. Pirozmand, P., Hosseinabadi, A.A.R., Farrokhzad, M., Sadeghilalimi, M., Mirkamali, S., Slowik, A.: Multi-objective hybrid genetic algorithm for task scheduling problem in cloud computing. Neural Comput. Appl. **33**(19), 13075–13088 (2021). https://doi.org/10.1007/s00521-021-06002-w
23. Sardjono, W., Priatna, W., Nugroho, D.S., Rahmasari, A., Lusia, E.: Genetic algorithm implementation for application of shifting work scheduling system. ICIC Exp. Lett. **15**(7), 791–802 (2021)
24. Shen, L., Zhang, G.: Optimization design of civil engineering construction schedule based on genetic algorithm. In: Journal of Physics: Conference Series, vol. 1852, p. 032055. IOP Publishing (2021)
25. Shuai, C.J.: Design of automatic course arrangement system for electronic engineering teaching based on monte carlo genetic algorithm. Secur. Commun. Netw. **2021** (2021)
26. Tang, J., Yang, Y., Hao, W., Liu, F., Wang, Y.: A data-driven timetable optimization of urban bus line based on multi-objective genetic algorithm. IEEE Trans. Intell. Transp. Syst. **22**(4), 2417–2429 (2020)
27. Tung Ngo, S., Jafreezal, J., Hoang Nguyen, G., Ngoc Bui, A.: A genetic algorithm for multi-objective optimization in complex course timetabling. In: 2021 10th International Conference on Software and Computer Applications, pp. 229–237 (2021)
28. Xie, L., Chen, Y., Chang, R.: Scheduling optimization of prefabricated construction projects by genetic algorithm. Appl. Sci. **11**(12), 5531 (2021)
29. Zaman, F., Elsayed, S., Sarker, R., Essam, D.: Hybrid evolutionary algorithm for large-scale project scheduling problems. Comput. Ind. Eng. **146**, 106567 (2020)