© Universiti Tun Hussein Onn Malaysia Publisher's Office

IJIE

http://penerbit.uthm.edu.my/ojs/index.php/ijie
ISSN : 2229-838X e-ISSN : 2600-7916

The International Journal of Integrated Engineering

Intelligent Electrical Device Load Scheduling for Building Energy Management

Casi Setianingsih[1*], Muhammad Ary Murti[1], Randy Erfa Saputra[1],Willy Mochamad Ilham[1], Mohammad Iqbal Nurhadi[1]

[1]School of Electrical Engineering,
Telkom University,Telekomunikasi Street, Bandung, 40257, INDONESIA

*Corresponding Author

DOI: https://doi.org/10.30880/ijie.2022.14.04.023
Received 14 July 2021; Accepted 17 April 2022; Available online 20 June 2022

Abstract: Currently, electrical energy is one of the primary needs in everyday life. Almost every device used for daily life uses electric power. However, the lack of user awareness in electronic devices can cause monthly electricity bills to be out of control, both in-office and residential areas. To improve the efficiency of electrical energy use in offices and housing, we designed a management system and control of daily electricity consumption so that it does not exceed the target of the monthly electricity bill. In this system, we integrate the priority of electronic devices into the optimization algorithm. This system uses a priority queue algorithm and genetic algorithm in scheduling electronic devices optimally. The system was created in the form of a website to limit the use of electrical energy by optimizing device scheduling. The method used is the Genetic Algorithm to calculate the duration of use of each electrical device so as not to exceed the predetermined monthly budget limit. This method will provide a recommended number of operating hours per day for 30 days for each electrical device that has been registered in the system. At the same time, the priority queue algorithm will adjust the duration of operation of each device based on the priority order of the device. Based on the test, the optimal fitness value is in the 60th generation, with an average execution time of 0.18 seconds. In testing the rules for both methods, this system has an accuracy rate of 100%. The system has been running according to the designed regulations.

Keywords: Web, database, energy management, genetic algorithm, priority queue algorithm, electronic loads

1. Introduction

Electrical energy has become a necessity for society; almost all devices used in everyday life require electrical power. Generally, the use of these electronic devices is still controlled manually [1][2]. To turn on and turn off an electronic device, users must interact directly with the device, such as pressing the power button, plugging, or disconnecting the cable to the socket. However, due to user negligence, sometimes some electronic devices that are not in use forget to turn them off. So that, without realizing it, can lead to an increase in monthly electricity bills or a waste of electricity quota [1][3].
Another example is in an office area. For example, electronic devices that are no longer in use will always turn on accidentally. That problem is caused by user negligence in using electronic devices unwisely. As a result, the monthly electricity bill exceeds the set budget.

To improve the efficiency of the use of electrical energy, a management system and load sharing of daily electronic devices are adjusted to the user's electricity cost target each month. In this system, the price per kWh of electricity is

**Corresponding author: setiacasie@telkomuniversity.ac.id*

adjusted according to the building class by referring to the provisions of the state electricity company in Indonesia. According to building type in Indonesia, the tariff for electricity usage has special rules according to the data applied by the Ministry of Energy and Mineral Resources in 2021. Therefore, in determining the cost of electricity consumption, each kWh corresponds to the class of the building. Building groups are divided into several types, namely small households, medium households, large households, medium businesses, large businesses, medium industries, large industries, and offices. The kinds of groups, of course, have different prices for each kWh according to the size of the building. Therefore, this building data collection is used to calculate the cost per kWh is more accurate and more accessible according to user needs [4].

The system was created in the form of a website to limit the use of electrical energy by optimizing device scheduling. The method used is the Genetic Algorithm to calculate the duration of use of each electrical device so as not to exceed the predetermined monthly budget limit. This method will provide a recommended number of operating hours per day for 30 days for each electrical device that has been registered in the system. Genetic algorithms can optimize complex problems and vast search space, so it is very suitable for cases with a wide variety of solutions, such as this case. A genetic algorithm is also one of the accurate optimization algorithms because in finding a solution, this algorithm will look for the most optimal value or repeat the calculation until it gets the best deal. In this study, there are 5 steps: initialization of the population with the data used is data from user input, chromosome evaluation to determine fitness values, chromosome selection to determine parental values, and the crossover is a cross between selected parents and the last is the mutation process. Finally, change the gene so that the gene value is not equal to zero [5][6].

In previous studies, device scheduling has been carried out to save electrical energy by adjusting the operating hours of household appliances to reduce costs to a minimum with price variations based on time. Several techniques have been used; the first is linear programming in scheduling electrical devices without considering uncertainty. The second is the stochastic scheduling technique modeling stochastic energy consumption patterns on electrical devices used [7]. Furthermore, using mathematical calculations, the home energy management system (HEMS) divides the types of electrical device categories into interruptable/uninterruptible and adjustable/unregulated power.

Using the Mixed-Integer Nonlinear (MINL) approach, this system can perform optimal scheduling by considering the level of user comfort [8]. The following system is real-time electrical scheduling (RTES) for managing the use of electrical energy in smart homes. The system aims to minimize cost payments by optimally scheduling intelligent appliances with a rolling 24-hour horizon using genetic algorithms but without the limitation of monthly electricity bill targets and electricity tariff differences for each building [9].

The solution for this case, we created a web-based software system that allows users to manage, monitor, and control the use of electrical energy automatically using priority queue algorithms and genetic algorithms. Thus, electricity consumption can be more efficient and controlled, and the monthly electricity bill target can be met. Furthermore, scheduling electrical devices from this method will be sent to Antares as an IoT Platform and forwarded to every connected IoT device.

2. Methodology

Figure 1 shows the system overview in this research. The system is designed to send data to Antares rather than passing it on to the hardware. Data from the database will be displayed first through the website; then, the data will be processed with Genetic Algorithms using the Python programming language. Then the results from the Genetic Algorithm will be saved into the database and displayed again on the website. Finally, the results from the system already stored in the database will be sent to Antares. Website made for controlling used electronic devices. A Genetic algorithm is used to calculate and provide solutions parameters that the user has entered the website. For example, this method will provide recommended hours of use for each electronic device for each day inputted to fulfill the amount of electricity bill desired by the user.

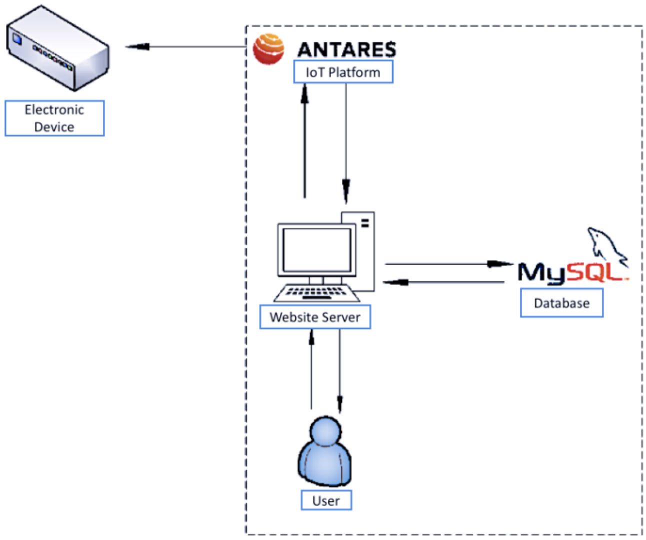

Fig. 1 - General system overview

2.1 Priority Queue Algorithms

A priority queue is a form of a data structure based on queue structure [10]. The priority queue has three types of priority, namely high, medium, and low [11]. A priority queue is generally followed by several sets of instructions, such as insert and delete, and looks for a maximum or minimum value [12]. This algorithm can be run in parallel to achieve high execution time speeds of a data structure [13]. The priority queue algorithm works based on the Higher Priority In First Out (HPIFO) principle, where the work with the highest priority will be completed first [14]. There are two types of priority queue algorithms, namely:

- Ascending Priority Queue, where data is sorted with increasing priority.
- Descending Priority Queue, where data is sorted with decreasing priority.

In addition, there are two fundamental operations used to change data in the priority queue algorithm, namely:

- Enqueue, namely the process of adding data to the end of the data sequence [15][16].
- Dequeue, which is the process of deleting data at the beginning of a data sequence [15][16].

The system flowchart, as shown in Figure 2, explains how this system works. First, the user will enter electronic device data and token data on the website. Next, data that the user from the website has entered into the database will be displayed. Then the data is processed with the priority queue algorithm using the Python programming language. In this algorithm, the available token data will be compared with the power requirements of each device; if the power is found, then the device will be given a status. Otherwise, the system will turn off the lower priority device then will compare the token data back with the device power requirements. If the token is fulfilled, then the device will be given the status on; if it still does not complete, then the device will be given the off status. Then the results of the status of each device will be saved into the database and displayed again on the website. Finally, the results from the system already stored in the database will be sent to Antares. This process is run every 24 hours.

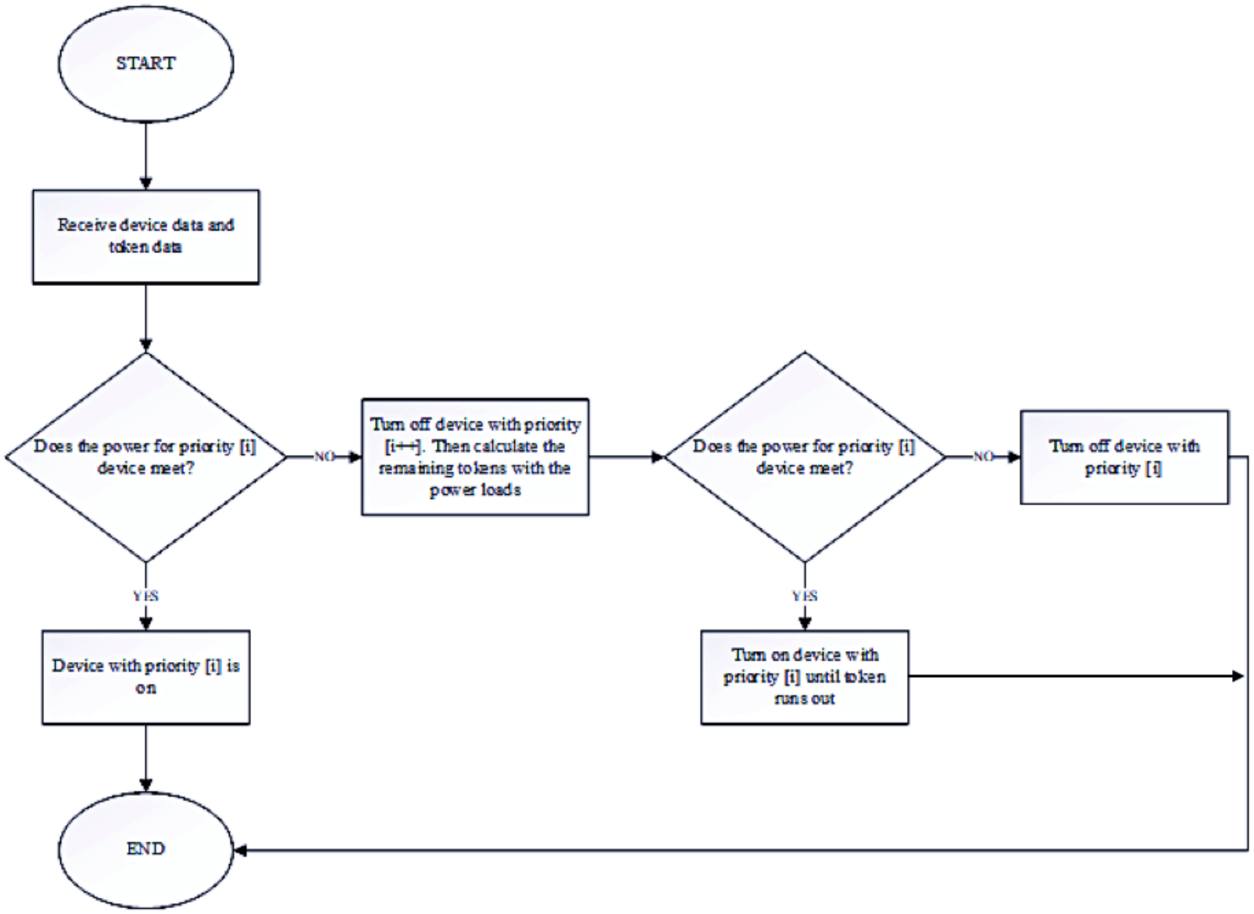

Fig. 2 - Priority queue flowchart

Table 1 - Rules system

Indicator	Indicator Variable	Rule	Rule Variable
Token meets power requirements	A	IF A AND C THEN M	A1
Token does not meet power requirements	B	IF B AND C THEN N	A2
Power loads P1	C	IF A AND D THEN O	A3
Power loads P1+P2	D	IF B AND D THEN P	A4
Power loads P1+P2+P3	E	IF A AND E THEN Q	A5
Power loads P1+P2+P3+P4	F	IF B AND E THEN R	A6
Power loads P1+P2+P3+P4+P5	G	IF A AND F THEN S	A7
Power loads P1+P2+P3+P4+P5+P6	H	IF B AND F THEN T	A8
Power loads P1+P2+P3+P4+P5+P6+P7	I	IF A AND G THEN U	A9
Power loads P1+P2+P3+P4+P5+P6+P7+P8	J	IF B AND G THEN V	A10
Power loads P1+P2+P3+P4+P5+P6+P7+P8+P9	K	IF A AND H THEN W	A11
Power loads P1+P2+P3+P4+P5+P6+P7+P8+P9+P10	L	IF B AND H THEN X	A12
Device with priority 1 turn on	M	IF A AND I THEN Y	A13
Device with priority 1 turn off	N	IF B AND I THEN Z	A14
Device with priority 2 turn on	O	IF A AND J THEN AA	A15
Device with priority 2 turn off	P	IF B AND J THEN BB	A16
Device with priority 3 turn on	Q	IF A AND K THEN CC	A17
Device with priority 3 turn off	R	IF B AND K THEN DD	A18

Device with priority 4 turn on	S	IF A AND L THEN EE	A19
Device with priority 4 turn off	T	IF B AND L THEN FF	A20
Device with priority 5 turn on	U		
Device with priority 5 turn off	V		
Device with priority 6 turn on	W		
Device with priority 6 turn off	X		
Device with priority 7 turn on	Y		
Device with priority 7 turn off	Z		
Device with priority 8 turn on	AA		
Device with priority 8 turn off	BB		
Device with priority 9 turn on	CC		
Device with priority 9 turn off	DD		
Device with priority 10 turn on	EE		
Device with priority 10 turn off	FF		

Based on table 1 and figure 2, this system has several rules used to decide each electronic device's status. In addition, some of the rules above are entered into the expert system used to determine decisions from the data read in the database.

2.2 Genetic Algorithms

The genetic algorithm is one of the optimization algorithms used to obtain a solution. This algorithm can be implemented in various problems and give better results for each iteration until it is close to optimal [17]. Figure 3 shows the sequence of processes in this system. First, the system retrieves device data and data on the number of bills and how many days the user wants from the database. After the desired data is obtained, it will then be optimized using a Genetic Algorithm that has 5 stages, namely population initialization, chromosome evaluation, chromosome selection, crossover, and mutation, which will be discussed in more detail in Figure 4. Then the system will issue recommendations for the use of hours every day for each electronic device. The optimization results are then stored in a database, and this data will be displayed on the website page and uploaded to the Antares server.

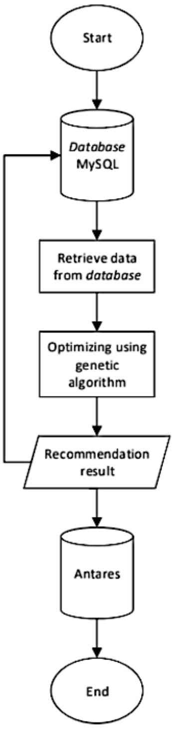

Fig. 3 - System flow diagram

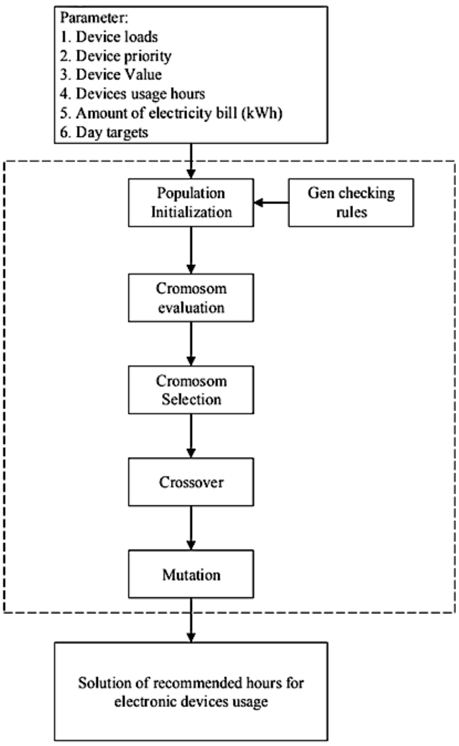

Fig. 4 - Steps of genetic algorithm

The explanation of Figure 4 can be seen below.

A. Population Initialization

The implementation of the genetic algorithm begins with initializing a population of chromosomes which are usually random. Population initialization determines the number and length of chromosomes in the population to be used [18][19]. In this study, every 1 chromosome has several genes depending on a large amount of data on electronic devices; the formula is as follows:

$$P = K \times jD \tag{1}$$

P = Population.
K = Chromosomes with the same length as the number of electronic device data.
jD = amount of electronic device data.

Then to get the threshold value for the Genetic Algorithm, you can use the formula below. The amount has been converted into kWh form, and the days you want to spend, so the number of kWh has been inputted.

$$Th = Pc / H \tag{2}$$

Th = Threshold.
Pc = The amount of the bill that has been converted into kWh.
H = number of days.

Provisions :

- Priority device 1 = Random gene value range 18-24.
- Priority device 2 = Random gene value range 16-22.
- Priority device 3 = Random gene value range 14-20.
- Priority device 4 = Random gene value range 12-18.
- Priority device 5 = Random gene value range 10-16.
- Priority device 6 = Random gene value range 8-14.
- Priority device 7 = Random gene value range 6-12.
- Priority device 8 = Random gene value range 4-10.
- Priority device 9 = Random gene value range 2-8.
- Priority device 10 = Random gene value range 1-6.

B. Chromosome Evaluation

Chromosome evaluation is the process of calculating the fitness value obtained from adding up the values of each chromosome with the following formula [20][21].

$$Fitness = \sum_{i=1}^{n} G_i V_i \tag{3}$$

n = Chromosome length.
G_i = The i-th gene.
V_i = The i-th value.

Provisions:

- Priority device 1 = Value 1000.
- Priority device 2 = Value 900.
- Priority device 3 = Value 800.
- Priority device 4 = Value 700.
- Priority device 5 = Value 600.
- Priority device 6 = Value 500.
- Priority device 7 = Value 400.
- Priority device 8 = Value 300.
- Priority device 9 = Value 200.
- Priority device 10 = Value 100.

C. Chromosome Selection

Chromosome selection is made to get chromosomes that have the potential to become parents for the following process. Chromosome selection can be made in several ways, namely Roulette Wheel Selection and Tournament Selection [22]. Roulette Wheel Selection is made based on the probability of each chromosome [23]. The genes in the chromosomes will vary greatly depending on the fitness value. Selection is made by generating random values from all fitness values from the population that has been evaluated. Meanwhile, the Tournament Selection is made to get the best parents at random, which is very dependent on fitness values. Selection begins by bringing up the fitness value of the population that has been evaluated, then the chromosome with the most significant fitness value is selected.

D. Crossover

The results from the chromosome selection are used as parents for the crossover process. This process will produce two new chromosomes from each pair of parents. The crossover also has various methods, namely One Point Crossover and Multi-Point Crossover [24][25]. One Point Crossover, swapping the genes of each pair of parents with one crossing point. Multi-point crossover, meanwhile, switches the genes of each pair of parents with multiple crossing points [26]. Finally, a crossover is carried out between two chromosomes, and the result is two new chromosomes. In this stage, there is a crossover probability that is used to determine whether a crossover needs to be done or not. The value of this probability is independent because there is no certainty about the best possibility. A random value will be the reference for the crossover iteration; this process will stop if the generated random value is less than the crossover probability value.

E. Mutation

The mutation process is carried out by exchanging genes in the chromosome with the following gene if it does not meet the requirements [27][28]. Mutations are like crossovers; mutations are not always carried out, and there is also a mutation probability for which there is no definite rule regarding the value. The process carried out is changing the value of the gene to ensure that no gene contains a value of 0, replacing genes that have a value of 0 with a random value of 1 to 24.

Table 2 - Rules for determining the distance of random values

Indicator	Indicator Variable	Rule	Rule Variable
Device P1	A	IF A AND K THEN M	A1
Device P2	B	IF B AND K THEN M	A2
Device P3	C	IF C AND K THEN M	A3
Device P4	D	IF D AND K THEN M	A4
Device P5	E	IF E AND K THEN M	A5
Device P6	F	IF F AND K THEN M	A6
Device P7	G	IF G AND K THEN M	A7
Device P8	H	IF H AND K THEN M	A8
Device P9	I	IF I AND K THEN M	A9
Device P10	J	IF J AND K THEN M	A10
24 Hours	K	IF A AND L THEN N	A11
Autogenerated Hours	L	IF B AND L THEN O	A12
Gene Value 24	M	IF C AND L THEN P	A13
Random gene value range 18-24	N	IF D AND L THEN Q	A14
Random gene value range 16-22	O	IF E AND L THEN R	A15
Random gene value range 14-20	P	IF F AND L THEN S	A16
Random gene value range 12-18	Q	IF G AND L THEN T	A17
Random gene value range 10-16	R	IF H AND L THEN U	A18
Random gene value range 8-14	S	IF I AND L THEN V	A19
Random gene value range 6-12	T	IF J AND L THEN W	A20
Random gene value range 4-10	U		
Random gene value range 2-8	V		
Random gene value range 1-6	W		

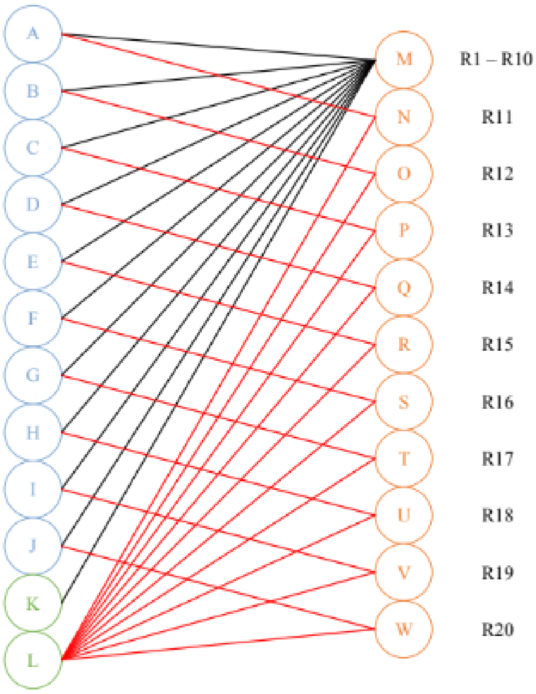

Fig. 5 - Decision tree

Figure 5 shows the decision-making rules obtained from the knowledge base in table 2, then presented in a decision tree. This rule will be a reference for determining the initial chromosome at the population initialization step.

3. Results and Discussion

3.1 Priority Queue Algorithms

The rules test aims to determine whether the rules used in the priority queue algorithm are already running according to the system design.

Table 3 - The testing result of priority queue algorithm rules

Test	Rule	Variable		Data Sent
		Total Loads (KWh)	Token (KWh)	
1	A1	21,6	60	Status P1: ON
2	A3	29,6	60	Status P2: ON
3	A5	39,2	60	Status P3: ON
4	A7	39,6	60	Status P4: ON
5	A9	42,6	60	Status P5: ON
6	A11	47,4	60	Status P6: ON
7	A13	52,92	60	Status P7: ON
8	A15	53,1	60	Status P8: ON
9	A17	54,9	60	Status P9: ON
10	A19	57,3	60	Status P10: ON
11	A2	21,6	2,7	Status P1: OFF
12	A4	29,6	2,7	Status P2: OFF
13	A6	39,2	2,7	Status P3: OFF
14	A8	39,6	2,7	Status P4: OFF
15	A10	42,6	2,7	Status P5: OFF

16	A12	47,4	2,7	Status P6: OFF
17	A14	52,92	2,7	Status P7: OFF
18	A16	53,1	2,7	Status P8: OFF
19	A18	54,9	2,7	Status P9: OFF
20	A20	57,3	2,7	Status P10: OFF

Based on table 3, the input data is tested, and the expected results have been met; this shows that the priority queue algorithm testing has fulfilled the system design with 100% accuracy. The devices entered in the system database are flexible; the system can accommodate up to 10 electronic devices. It can be seen in Table 3 that testing was carried out with 10 electronic device data entered, and the input token was 60 KWh. Owned tokens will be compared with the power requirements of each device sequentially based on their priority; then, the token will be reduced when the algorithm has decided the device is given a status On. For further comparison with the power requirements of the device, a reduced token will be used.
Execution time testing aims to find out how long it takes the system to run the entire process.

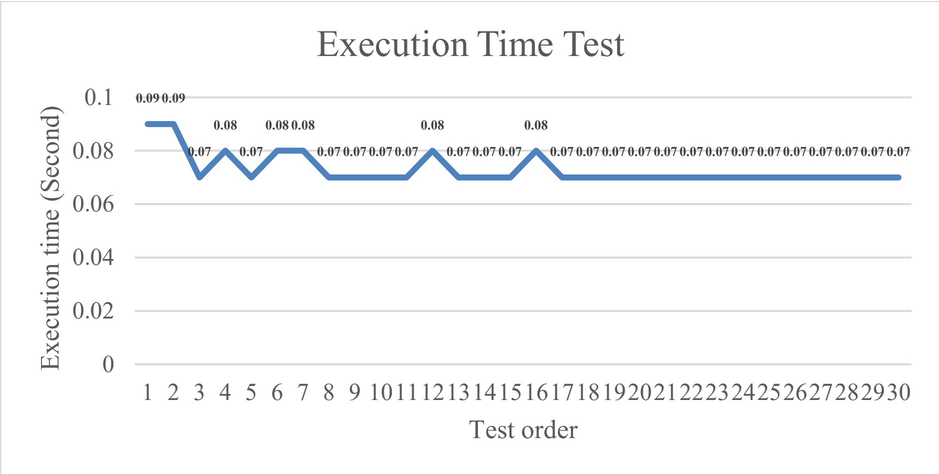

Fig. 6 - Execution time result

From Figure 6, the results of the execution time test above. It can be seen that the time needed for the system to determine the device's status using the priority queue algorithm is between 0.07 seconds to 0.09 seconds, and the average time required is 0.073 seconds. These results are performed by running the algorithm 30 times.

3.2 Genetic Algorithms

Input Value :

- Priority device 1 : Power = 0.9 kWh, Priority = 1, Value = 1000 Hours = Auto.
- Priority device 2 : Power = 1 kWh, Priority = 2, Value = 900 Hours = Auto.
- Priority device 3 : Power = 1.2 kWh, Priority = 3, Value = 800 Hours = Auto.
- Priority device 4 : Power = 0.05 kWh, Priority = 4, Value = 700 Hours = Auto.
- Priority device 5 : Power = 0.125 kWh, Priority = 5, Value = 600 Hours = Auto.
- Priority device 6 : Power = 0.6 kWh, Priority = 6, Value = 500 Hours = Auto.
- Priority device 7 : Power = 0.23 kWh, Priority = 7, Value = 400 Hours = Auto.
- Priority device 8 : Power = 0.09 kWh, Priority = 8, Value = 300 Hours = Auto.
- Priority device 9 : Power = 0.15 kWh, Priority = 9, Value = 200 Hours = Auto.
- Priority device 10 : Power = 0.3 kWh, Priority = 10, Value = 100 Hours = Auto.
- The Amount of The Bill: Rp.1,000,000 (681.53320 kWh).
- Day: 7 Days.

A. Population Initialization
P = K x jD
P = 10 device data x 10 device data
P = 100 gene

Following the calculation of the number of genes above, 100 genes will be used. For example, if the clock input is 24, the device must be on for 24 hours, then the gene for that device will be filled in 24. However, if the clock enters Auto, the device will get a random gene according to its priority. Figure 7 shows an overview of the initial population.

```
Population size = (10, 10)
[[21 17 19 17 16 10 12 10  7  5]
 [23 22 20 18 12 13 10  9  8  4]
 [22 16 15 16 12 10 10  9  7  4]
 [20 16 15 17 10 12  8  9  2  1]
 [21 17 16 16 13 13  6  8  4  3]
 [20 17 14 17 12 10  6  5  2  3]
 [24 17 15 18 10 12  9  7  3  3]
 [23 17 17 18 16  9 10  6  4  6]
 [19 19 20 18 16  8  8  7  6  4]
 [18 21 19 16 15  9  8  9  7  5]]
```

Fig. 7 - Initial population

After that, a calculation is made for the threshold for the number of bills that have been converted to kWh and the days that have been inputted, and the total bill is then divided by days so that the kWh for each day is obtained, which will be used as the threshold as follows:

Th = Pc / H
Th = 681.53320 / 7
Th = 97 kWh

B. Chromosome Evaluation

After getting the initial population, each chromosome will be calculated as the fitness value. This calculation is only for the first chromosome, and then this process will be carried out for each chromosome.

$$Fitness = \sum_{i=1}^{n} G_i V_i$$

Fitness = (21x1000) + (17x900) + (19x800) + (17x700) + (16x600) + (10x500) + (12x400) + (10x300) + (7x200) + (5x100)
Fitness = 87700

C. Chromosome Selection

At this stage, a selection is made to get the best parents at random. The step taken is to divide the total population by 2, then the system will automatically select the chromosomes with the most significant fitness value to become parents. Figure 8 shows the result of the Parent selection step.

```
Parents:
[[23. 22. 20. 18. 12. 13. 10.  9.  8.  4.]
 [21. 17. 19. 17. 16. 10. 12. 10.  7.  5.]
 [23. 17. 17. 18. 16.  9. 10.  6.  4.  6.]
 [19. 19. 20. 18. 16.  8.  8.  7.  6.  4.]
 [18. 21. 19. 16. 15.  9.  8.  9.  7.  5.]]
```

Fig. 8 - Parent selection results

D. Crossover

After getting the parents, a crossover process is carried out, which is crossing between each parent which will produce offspring or new chromosomes.

Parents A	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	23	22	20	18	12	13	10	9	8	4
Parents B	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	21	17	19	17	16	10	12	10	7	5
Hasil										
Parents A	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	23	21	24	17	13	14	9	10	7	5
Parents B	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	22	20	24	13	16	11	12	9	8	4

Fig. 9 - Crossover illustration

The crossover process is repeated until the generated random value is smaller than the crossover probability [29]. The crossover probability used in this study is 0.4. The selection of the crossover probability is obtained from the experiment, with the result that the optimal fitness value is found in the crossover probability of 0.4. After the crossover process is complete, each chromosome is checked to ensure the total gene value on the chromosome does not exceed the threshold value. If it exceeds the threshold, then the value of the gene on that chromosome will be changed to zero, starting from the last gene of each chromosome until the total gene value is equal to or less than the threshold. Figure 9 shows the Crossover process, and figure 10 shows the result of the Crossover process step.

```
Crossover:
[[19. 17. 16. 17. 14. 11.  6.  8.  8.  4.]
 [18. 16. 17. 16. 11. 12. 10.  8.  3.  3.]
 [18. 21. 15. 12. 11.  8.  7.  4.  4.  5.]
 [21. 22. 19. 12. 12. 13. 11.  6.  3.  5.]
 [21. 17. 20. 16. 15.  9.  8.  4.  4.  5.]]
```

Fig. 10 - Crossover result chromosomes

E. Mutation

The last stage is mutation. This process is carried out automatically by the system by optimizing the value of existing genes, such as replacing a gene with a value of 0 with a gene with a random value from a distance of 1 to 24. Like a crossover, this process is repeated until more random values are generated, smaller than the mutation probability. In this study, the mutation probability used is 0.4. The same as the selection of the crossover probability, the choice of the mutation probability is also obtained from the experiment. The result is that the optimal fitness value is found in the crossover probability of 0.4. Furthermore, each chromosome is checked to ensure the total gene value on the chromosome does not exceed the threshold. If it exceeds, then the value of the gene on the chromosome will be changed to zero starting from the last gene of each chromosome until the value is equal to or lighter than the threshold.

```
Mutation:
[[19. 17. 16. 17. 14. 11.  5.  8.  8.  4.]
 [18. 16. 17. 16. 11. 12. 10.  8.  3.  3.]
 [18. 21. 15. 12. 11.  8.  7.  4.  4.  5.]
 [21. 22. 19. 12. 12. 13. 11.  6.  3.  5.]
 [21. 17. 20. 16. 15.  9.  8.  4.  4.  5.]]
```

Fig. 11 - Chromosome mutation results

The chromosomes resulting from this mutation will be recombined with the chromosomes resulting from the selection process [30].

```
Last generation:
[[23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 22 21 17 23 22  4]
 [23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 21 21 17 22 22  7]
 [23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 22 21 17 22 22  7]
 [23 24 20 18 22 21 17 23 22  4]]
```

Fig. 12 - Final population

The process from the first to the last stage will continue to be carried out until the best fitness value is produced [31]. Then, chromosomes whose fitness and total gene values do not exceed the threshold will be selected as solutions. Figure 11 shows the result of the Mutation process step, and figure 12 shows a final population of 1st generation.

```
Last Weight:
95.42999999999999

Fitness of the last generation:
[115400 115400 115400 115400 115400 115400 114800 115400 115400 115400]

The optimized parameters for the given inputs are:
[array([23, 24, 20, 18, 22, 21, 17, 22, 22,  7])]
Device :  23
Device :  24
Device :  20
Device :  18
Device :  22
Device :  21
Device :  17
Device :  22
Device :  22
Device :  7
```

Fig. 13 - Chromosome solutions

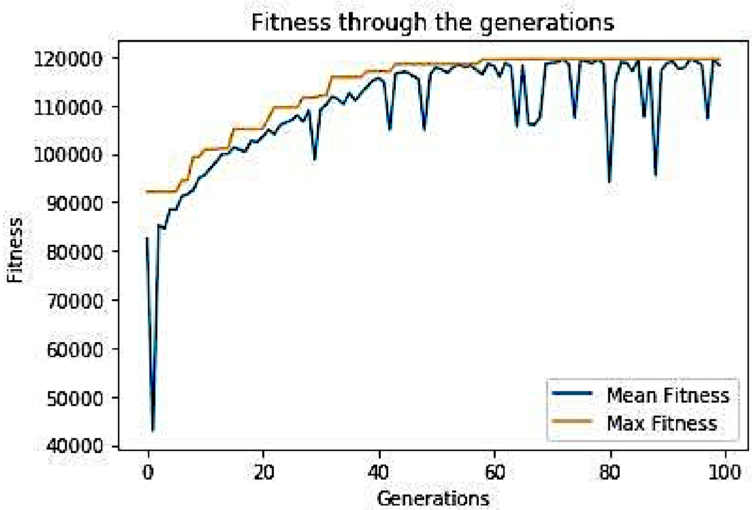

Fig. 14 - Fitness value graph

Figure 13 describes the system outcomes, and figure 14 shows the fitness value testing for this system. The optimal fitness value is obtained in generation 60; this can be ascertained because there is no further increase in fitness values until generation 100. This is because the fitness value of the genetic algorithm has reached convergence in the 60th generation, so it can be interpreted that the system has found the most optimal chromosome or solution globally [32].

Table 4 - Rules Validation

Test	Rule	Variable		Gene
		Device	Hour	
1	R1	Priority 1	24	24
2	R2	Priority 2	24	24
3	R3	Priority 3	24	24
4	R4	Priority 4	24	24
5	R5	Priority 5	24	24
6	R6	Priority 6	24	24
7	R7	Priority 7	24	24
8	R8	Priority 8	24	24
9	R9	Priority 9	24	24
10	R10	Priority 10	24	24
11	R11	Priority 1	Auto	24
12	R12	Priority 2	Auto	20
13	R13	Priority 3	Auto	18
14	R14	Priority 4	Auto	17
15	R15	Priority 5	Auto	12
16	R16	Priority 6	Auto	10
17	R17	Priority 7	Auto	9
18	R18	Priority 8	Auto	5
19	R19	Priority 9	Auto	3
20	R20	Priority 10	Auto	4

Based on Table 4, it is shown that the rule validation test is successful 100%, and the algorithm will run properly.

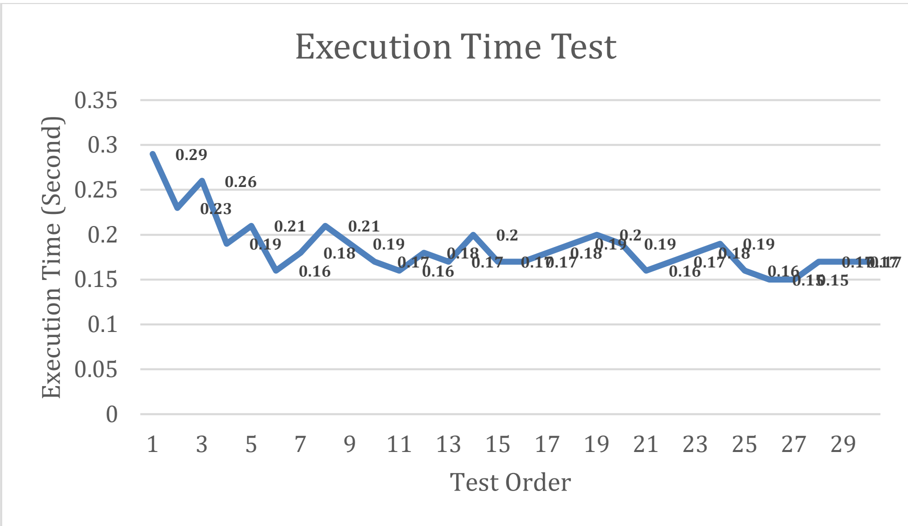

Fig. 15 - Method execution time

Figure 15 shows that the time needed by the system to carry out the entire process is between 0.15 seconds to 0.3 seconds, with an average execution time of 0.18 seconds.

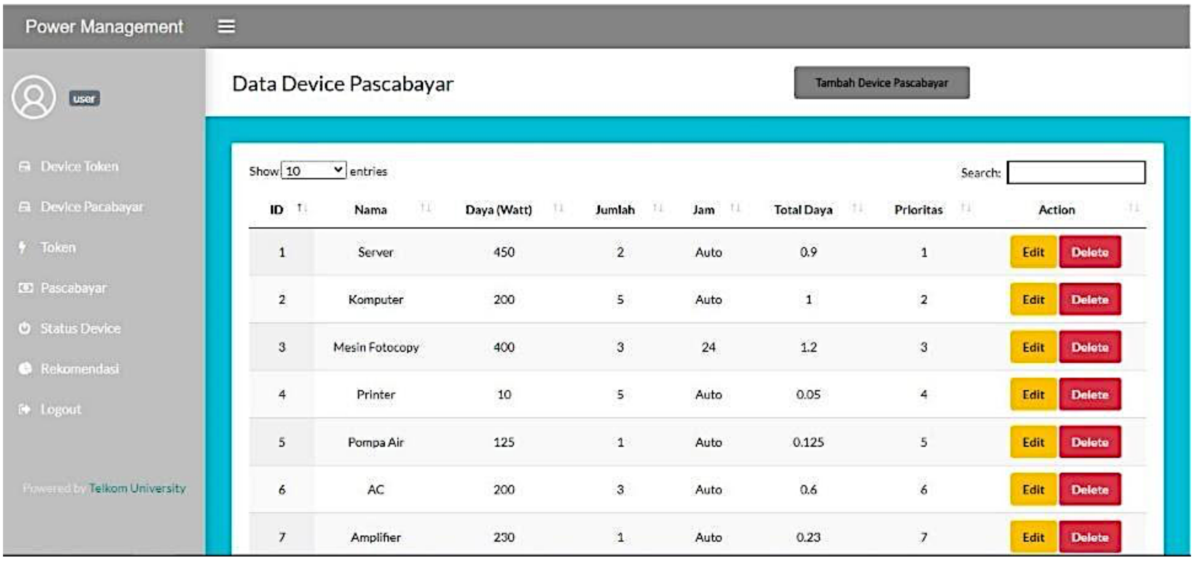

Fig. 16 - Postpaid data device page

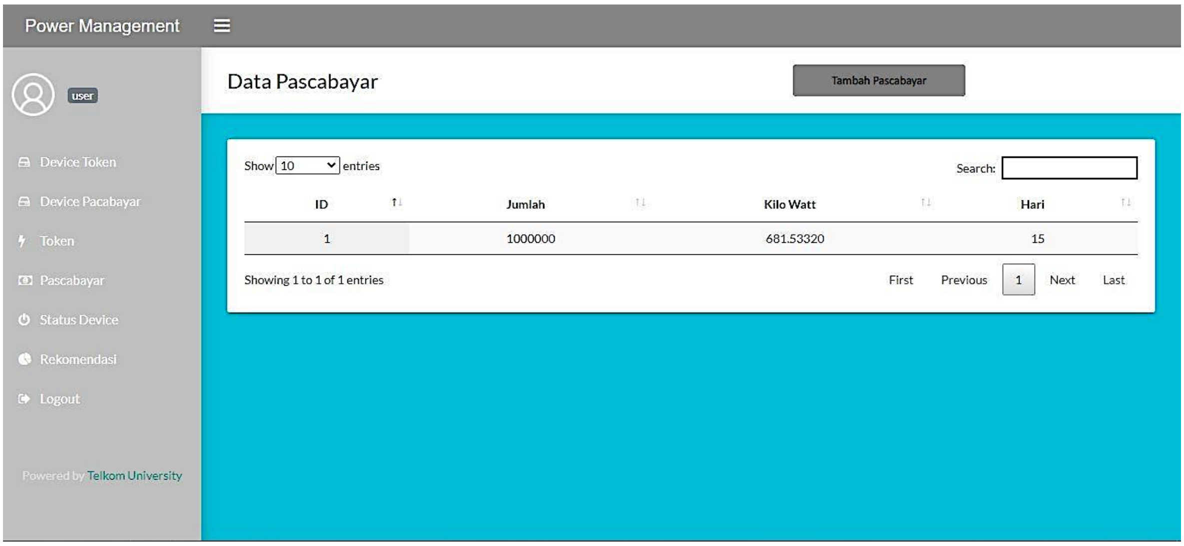

Fig. 17 - Postpaid data page

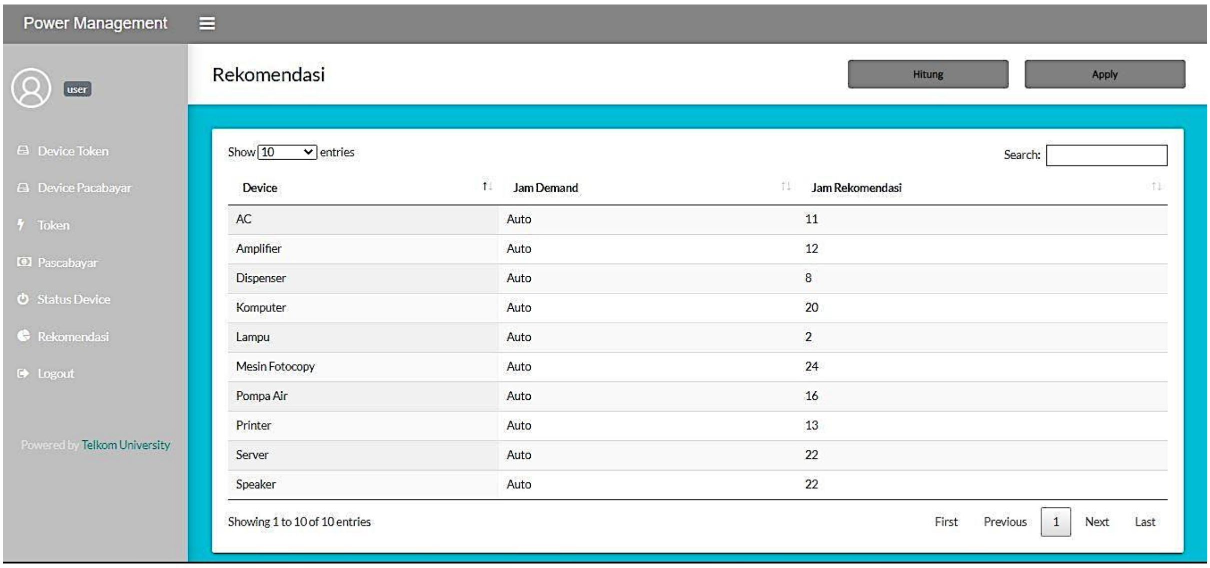

Fig. 18 - Recommendation page

Figure 16 is a website display covering an electronic device data page with a sub-menu of adding and editing data. Figure 17 is a postpaid page to fill in the desired monthly billing budget with the sub-menu data. Figure 18 is a recommendation page display which is a menu for displaying system scheduling results.

4. Conclusion

Based on the results of the tests and analyzes that have been carried out, it can be concluded that the Power Management website can display recommendation data on hours of use of electronic devices that have been inputted by the user, using a Genetic Algorithm. The optimal fitness value is in the 60th generation based on optimal fitness testing, and based on the rule test results, the accuracy level obtained is 100%. For the execution time testing, the Genetic Algorithm is required to run the system, starting from receiving data from the database to sending data back to the database again; the average execution time is 0.18 seconds. At the same time, the priority queue algorithm succeeds in selecting and turning off electronic devices automatically according to their priority when the token is not fulfilled to turn on all devices simultaneously. Based on the results of testing the rules, the level of accuracy obtained is 100%. Furthermore, the test execution process time results, the time required by the system to determine the status of an electronic device, can be completed in an average time of 0.073 seconds.

Acknowledgement

The author would like to acknowledge the Telkom University and LPDP. This study is financially supported by the grant RISPRO 2022 received from the LPDP.

References

[1] Widhiono, S. A., Murti, M. A., & Setianingsih, C. (2019, December). Electronic Loads Control and Management Using Priority Queue Algorithm on Android Based Smartphone. In 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) (pp. 154-159). IEEE.

[2] Wang, X., & Li, J. (2018, May). Design of Intelligent Home Security Monitoring System Based on Android. In 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC) (pp. 2621-2624). IEEE.

[3] Ozturk, H. K., Ceylan, H., Canyurt, O. E., & Hepbasli, A. (2005). Electricity estimation using genetic algorithm approach: a case study of Turkey. Energy, 30(7), 1003-1012.

[4] The Ministry of Energy and Mineral Resources's newest 2021 electricity tariff list [KWh price]. (2021).

[5] Babatunde, O. H., Armstrong, L., Leng, J., & Diepeveen, D. (2014). Zernike moments and genetic algorithm: Tutorial and application.

[6] Yan, G., Xie, G., Chen, Z., & Xie, K. (2008, May). Knowledge-based genetic algorithms. In International Conference on Rough Sets and Knowledge Technology (pp. 148-155). Springer, Berlin, Heidelberg.

[7] Chen, X., Wei, T., & Hu, S. (2013). Uncertainty-aware household appliance scheduling considering dynamic electricity pricing in smart home. *IEEE Transactions on Smart Grid*, *4*(2), 932-941.
[8] Huang, G., Yang, J., & Wei, C. (2016, October). Cost-Effective and comfort-aware electricity scheduling for home energy management system. In *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)(BDCloud-SocialCom-SustainCom)* (pp. 453-460). IEEE.
[9] Li, S., Yang, J., Song, W., & Chen, A. (2018). A real-time electricity scheduling for residential home energy management. *IEEE Internet of Things Journal*, *6*(2), 2602-2611.
[10] Dragicevic, K., & Bauer, D. (2008, April). A survey of concurrent priority queue algorithms. In 2008 IEEE International Symposium on Parallel and Distributed Processing (pp. 1-6). IEEE..
[11] Ambigavathi, M., & Sridharan, D. (2018). Energy efficient and load balanced priority queue algorithm for Wireless Body Area Network. Future Generation Computer Systems, 88, 586-593.
[12] Ioannou, A., & Katevenis, M. G. (2007). Pipelined heap (priority queue) management for advanced scheduling in high-speed networks. IEEE/ACM Transactions on Networking, 15(2), 450-461.
[13] He, X., Agarwal, D., & Prasad, S. K. (2012, December). Design and implementation of a parallel priority queue on many-core architectures. In 2012 19th International Conference on High Performance Computing (pp. 1-10). IEEE.
[14] De Gier, J., & Finn, C. (2014). Exclusion in a priority queue. Journal of Statistical Mechanics: Theory and Experiment, 2014(7), P07014.
[15] Kumar, N. C., Vyas, S., Cytron, R. K., Gill, C. D., Zambreno, J., & Jones, P. H. (2014). Hardware-software architecture for priority queue management in real-time and embedded systems. *International Journal of Embedded Systems*, *6*(4), 319-334.
[16] Tang, W. T., Goh, R. S. M., & Thng, I. L. J. (2005). Ladder queue: An O (1) priority queue structure for large-scale discrete event simulation. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, *15*(3), 175-204.
[17] Song, W., & Park, S. C. (2006, September). Genetic algorithm-based text clustering technique. In International Conference on Natural Computation (pp. 779-782). Springer, Berlin, Heidelberg.
[18] Mukhopadhyay, D. M., Balitanas, M. O., Farkhod, A., Jeon, S. H., & Bhattacharyya, D. (2009). Genetic algorithm: A tutorial review. International journal of grid and distributed computing, 2(3), 25-32.
[19] Kumar, R. Knowledge based operation and problems representation in genetic algorithms.
[20] Babatunde, O. H., Armstrong, L., Leng, J., & Diepeveen, D. (2014). A genetic algorithm-based feature selection.
[21] Jia, H. Z., Nee, A. Y., Fuh, J. Y., & Zhang, Y. F. (2003). A modified genetic algorithm for distributed scheduling problems. *Journal of Intelligent Manufacturing*, *14*(3), 351-362.
[22] Hu, J., Sun, Y., & Qingzhen, X. (2010, June). Notice of Retraction: The theory and application of Genetic Algorithm. In *2010 International Conference on Computer and Communication Technologies in Agriculture Engineering* (Vol. 1, pp. 155-157). IEEE.
[23] Mirjalili, S. (2019). Genetic algorithm. In *Evolutionary algorithms and neural networks* (pp. 43-55). Springer, Cham.
[24] Watanabe, M., Ida, K., & Gen, M. (2005). A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem. *Computers & Industrial Engineering*, *48*(4), 743-752.
[25] Hou, S., Liu, Y., Wen, H., & Chen, Y. (2011, December). A self-crossover genetic algorithm for job shop scheduling problem. In *2011 IEEE International Conference on Industrial Engineering and Engineering Management* (pp. 549-554). IEEE.
[26] Ikeda, K., & Kobayashi, S. (2002, September). Deterministic multi-step crossover fusion: A handy crossover composition for GAs. In *International Conference on Parallel Problem Solving from Nature* (pp. 162-171). Springer, Berlin, Heidelberg.
[27] Chen, C., & WAN, N. (2002). Adaptive selection of crossover and mutation probability of genetic algorithm and its mechanism [J]. *Control Theory & Applications*, *1*, 41-43.
[28] Yong11, K. H. J. J. S. (2006). Improving crossover and mutation for adaptive genetic algorithm. *Computer Engineering and Applications*, *12*, 027.
[29] Fauzi, M. A., Hanuranto, A. T., & Setianingsih, C. (2020, October). Intrusion Detection System using Genetic Algorithm and K-NN Algorithm on Dos Attack. In *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)* (pp. 1-6). IEEE.
[30] Arif, F. N., Irawan, B., & Setianingsih, C. (2020, November). Optimization of Storage Allocation For Final Goods Warehouse Using Genetic Algorithm. In *2020 2nd International Conference on Electrical, Control and Instrumentation Engineering (ICECIE)* (pp. 1-6). IEEE.
[31] Aziz, A., Nasution, S. M., & Setianingsih, C. (2020). Analysis Simple Additive Weighting And Genetic Algorithm For Traffic Management System. *International Journal of Integrated Engineering*, *12*(7), 104-112.
[32] Sharapov, R. R., & Lapshin, A. V. (2006). Convergence of genetic algorithms. *Pattern recognition and image analysis*, *16*(3), 392-397.