Hybrid Genetic Algorithms and Simulated Annealing for Multi-trip Vehicle Routing Problem with Time Windows

Amalia Kartika Ariyani[1], Wayan Firdaus Mahmudy[2], Yusuf Priyo Anggodo[3]
[1,2]Department of Informatics, Faculty of Computer Science, Brawijaya University, Indonesia
[3]Data Scientist, Ilmuone Data, Indonesia

Article Info	ABSTRACT
	Vehicle routing problem with time windows (VRPTW) is one of NP-hard problem. Multi-trip is approach to solve the VRPTW that looking trip scheduling for gets best result. Even though there are various algorithms for the problem, there is opportunity to improve the existing algorithms in order gaining a better result. In this research, genetic algoritm is hybridized with simulated annealing algoritm to solve the problem. Genetic algoritm is employed to explore global search area and simulated annealing is employed to exploit local search area. Four combination types of genetic algorithm and simulated annealing (GA-SA) are tested to get the best solution. The computational experiment shows that GA-SA1 and GA-SA4 can produced the most optimal fitness average values with each value was 1.0888 and 1.0887. However GA-SA4 can found the best fitness chromosome faster than GA-SA1.

Corresponding Author:

Wayan Firdaus Mahmudy,
Faculty of Computer Science, Brawijaya University,
Veteran Road, Malang, Indonesia.
Email: wayanfm@ub.ac.id

1. INTRODUCTION

The growth of the number of tourists in tourist destinations located in Banyuwangi, East Java, Indonesia, is increasing as the infrastructure improvements by the government. With the increasing number of visits, there are some problems for tourists coming from out of town or from abroad. Some of them find it difficult to determine which sights to visit with the time limit they have set due to limited access to available information. They can not decide the trip based on their time limit with minimum cost and arrive to the destination on opening hours [1]. That problem usually called as scheduling problem. Scheduling problem is to make a schedule to visit tour places that total time of the schedule is balanced under many contrains [2]. Scheduling problem is complex problem because to get best solution must be balancing allocation the visit that to pass all contrains. The main focus of scheduling is to minimize the total duration of the visit [3]. In addition, the delay of scheduling during change the part is very important because get influence the tota duration of the scheduling [4]. The research focus to get maximum solution with having minimum cost as low as possible. The tourist must have a schedule trip. To solve this problem needs a certain model. Anggodo et al [1] had finished this problem with same data. It can solve the problem with vehicle routing problem approach. Vehicle routing problem is procedure for solving scheduling problem in vehicle route.

Vehicle routing problem with time windows is a combination schedule visiting many places with time constrained service provision [5]. VRPTW is used to find optimal route of vehicle which leave a centralized depot. The solution of route gets from geographically dispersed customers. Finding vehicle scheduling and routing plan and minimizing total route distance are the focus of the problem [6]. VRPTW is used in logistics manufacturing with the complexity problem. As many as constrain makes the problem as difficult as. In fact, to solve VRPTW problem is easy when just having one depot and few nodes but the problem will become complex when having many depots and nodes. In addition, the problem will be more

complex when be found the flexible path [7]. Based on that, VRPTW is a non-determinstic polynominal-time hard (HP-hard) problem [6]. Previous research that proposed Anggodo et al [1] have constrain day of trip the tourist. Every finishing the trip on once day the tourist must go back to the hotel or centralized depot. In this study, the tourist will visit for many days so the problem has many trips. It is called multi-trip VRPTW that difference problem with only VRPTW. Multi-trip VRPTW makes problem to be complex because only VRPTW without multi-trip just need one leave and go back to the centralized depot.

Most researchers have been growing interest to use combination and developing heuristics algorithm to solve VRPTW problem. The hybrid heuristic algorithm gets very strong searching and the solution quality of method is high [8]. Geoseiri & Ghannadpour [5] proposed goal approach to solve routing problem, building and implemented genetic algorithm (GA) to solve the problem. Recently, many researchers used genetic algorithm to solve the problem. Ursani et al [9] used genetic algorithm in routing problem framework. Genetic algorithm can get maximum solution for Solomon benchmark problem, research problem, and E-Commerce Supplier [7], [10], [11]. However, genetic algorithm has weakness in local searching solution. Solving the roblem must improve to effective and efficient searching maximum solution. Wang et al [12] proposed combination genetic algorithm with particle swarm optimition. The focus of Genetic algorithm finds global solution and particle swarm optimization search local solution. It is balance and excellence but needing more time for computation. Improving crossover genetic algorithm use cyclic shift crossover with hill-climbing mechanism to generate a child in initializiton proses [13]. It is good model to solve the problem but this model just the focus for searching local area because crossover modification makes change focus search area.

Based on it, we need model to solve which balance to search solution and fast computation. In fact, genetic algorithm shows one model can to get maximum solution with focus in global search. Previously, genetic algorithm had could implementation in this case [1]. Many procedures do to get good local search. In this study, adding combination one heuristic method which focus in local searching. Crossover modification is not done because can change focus genetic algorithm from global search to local search. Mahmudy [14], improved simulated annealing (SA) for routing problem gets the best solution and fast computation. Simulated annealing focus on local search, when using large data simulated annealing tends to get poor solution. In this research, we propose new combination model of genetic algorithm and simulated annealing (GA-SA) to solve the problem. The proposed method will test with other model [15]. Focus this research is building new combination of genetic algorithm and simulated annealing and compare all GA-SA combination to get maximum solution.

2. RELATED WORK

Previsious research about routing problem have done to get maximum solution. It is not new problem however very interesting to find the model which solving in many conditions. The same data had been finished by Anggodo [1], it was finished using genetic algorithm. It proposed one-cut-point crossover and reciprocal exchange mutation to get the solution. to get maximum solution it was used permutation model for chromosome. Previous studies, GA can solve routing problem [9]. That study proposed combination mutation operator and recombination operator for the genetics algoritm operator. In that problem, chromosome representation was used permutation model. GA was used to solve routing problem with many constrains i.e. vehicle with limited capacity, depot constrains, and time windows constrains [5]. that research got maximum result because the main focus was the techniques genetic algorithm that used best cost-best route crossover, sequence best mutation, and permutation model for chromosome representation. Based on previsous studies, GA is good method to find solution in global searching and the best chromosome representation for route problem is permutation model.

Routing problem will get maximum solution when have method which balance to find solution. It is focus on global and local search. GA can do finding solution in global search, so needing method can find in local search. Mahmudy [14] prosesed improved simulated annealing to solve routing problem. It got good solution with fast computation. Combination GA and SA will get best solution which rather than other method. GA do finding solution in focus global search and SA do finding solution in focus local search. To get better solution will be used adaptive chromose [5]. It is good method however to need one's procedure to combination GA & SA. Fatyanosa et al [15] had finished hybrid GA and SA to solve benchmarks function. In that research used extended intermediate crossover and random mutation for operator GA. In this study, we will use hybrid GASA which proposed by Fatyanosa. other than that, will proposed procedure hybrid GASA and compare with Fatyanosa's GASA method.

3. DESCRIPTION OF THE PROBLEM

Vehicle Routing Problem with Time Windows (VRPTW) is one of the most studied VRP issues. The concept of VRPTW is the development of VRP problems with the addition of new constraints that is a time windows. Thus, the service of each customer must be started and terminated based on the time windows that have been determined. If the vehicle distributes up before the customer's windows time, it is subject to the waiting points, while the distribution vehicle that comes after the customer's windows time expires, then it is subject to penalty points [16].

The purpose of VRPTW's optimization problem is to determine the shortest distance route to minimize travel cost and the number of vehicles without breaking the time windows and vehicle capacity. The journey starts from a single depot then goes to each of the scattered customers. The distribution vehicle must reach the customer's place between the customer's time windows. The total weight of the goods transported should not exceed the capacity of the vehicle. Vehicles that have finished visiting the customer node must return to the initial depot within the predefined time windows depot [17]. Multi-Trip Vehicle Routing Problem (MT-VRP) is one of the development of classic VRP problems in which a vehicle travels on multiple routes within a certain time frame [18]. The MT-VRP is a special vehicle route issue where each vehicle can serve more than one trip where each trip starts from the depot, passes several customers and ends up in the depot and the total customer demand should not exceed capacity [19]. Multi-Trip VRPTW applies MT-VRP concept but there is addition of time limit of windows in it.

In this study, hybridization of genetic algorithm and simulated annealing is used to solve the problem of Multi-Trip Vehicle Routing Problem with Time Windows which is about scheduling tourist route in Banyuwangi Regency. There are several conditions that limit the study.
a.   There are 19 tourist data and 14 hotel data used in this research
b.   Trips are set for 3 days and occupy only 1 hotel as depot
c.   Travel is assumed using the car and road conditions smoothly
d.   The round trip time is assumed to be the same
e.   Time windows tourists start at 05.00-19.00
f.   Fitness value derived from total travel time and total tourist attraction can be visited within 3 days
g.   The more tourist objects and the less travel time required will result in greater fitness value
h.   Penalty value is the total value of violation against time windows tourist object and time windows tourists

4. ALGORITHM USED

4.1. Genetic Algorithm

Genetic Algorithm is one of search and optimization algorithms based on natural selection and genetic mechanisms that have been developed as effective optimization approaches to solve complex problems [20]. Genetic algorithm approach can help find a good solution for complex mathematical problems such as the VRP problem [21]. Multi-Trip Problems VRPTW is a combinatorial optimization problem by forming the appropriate reprepentation solution that will get optimal solution in the use of genetic algorithm for Multi-Trip VRPTW problem. In this study, we use procedure of genetic algorithm that is shown in Figure 1 [1].

```
Initialization parameter GA
GA (stopping condition) {
        Calculating fitness of chromosome
        // doing reproduction to get new chromosome
        Crossover using One-Cut-Point Crossover
        Mutation using Resiprocal Exchange Mutation
        Selection using Replacement Selection
}
```

Figure 1. Procedure of genetic algorithm

4.1.1. Chromosom Representation

In case about VRPTW, chromosome is an encode of destinations permutation that are visited by tourist. Using permutation will include all of tourism destinations so genetic algoritm can search optimum solution. Chromosom representation can be seen in Figure 2.

| 8 | 17 | 1 | 12 | 2 | 3 | 14 | 10 | 7 | 19 | 13 | 4 | 5 | 15 | 18 | 16 | 9 | 6 | 11 |

Figure 2. Chromosome representation

After getting the chromosome results, next step is to calculate the fitness value of the chromosome that has been formed. The fitness calculation begins as if it inserts a hotel index on a chromosome. In the first gene of the chromosome inserted the index of the selected hotel, for example the index of the selected hotel is 4, then the hotel index 4 is inserted at the beginning of the chromosome. Once the hotel index is inserted, the next step is to insert the tourist object gene one by one on the chromosome, if the total duration of the tourist visit has exceeded 14.45 then the insertion of the hotel index 4 returns. After the insertion of the hotel index the process of entering the return tourist gene is done with the same process until the day limit is entered. In this study it is assumed that the trip is done within 3 days so that the process of entering the chromosome gene is done in 3 trips. If the process of entering the gene has been completed, then the end of the chromosome is re-inserted the hotel index 4.

The calculation of fitness value is influenced by the parameters of travel time (*time*), *penalty*, and number of tourist destinations that can be visited. In the time parameter the greater the value of time, the smaller the fitness value obtained so equation 1 is used divided by 1 + time, the addition of value 1 in the time parameter is used to avoid infinity value if there is an occurrence of time value equal to 0. In the increasing penalty parameter penalty value will reduce the fitness value so that the penalty equation is used multiplied by the value -1 for the penalty value to be minus, if there is no penalty then the penalty value equal to 0. In the parameter of the number of attractions, the less number of visited tourist sites will reduce the value of fitness so that the value of the number of tourist attractions divided by the number of ideal tourist attraction that can be visited within 3 days as many as 10 attractions [1]. In this study Equation 1 shows the function to calculate the fitness value. The example of chromosome's fitness calculation can be seen in Table 1.

$$fitness = \frac{1}{1+time} + (penalty \ x - 1) + \frac{number \ of \ tourist \ destination}{10}, \qquad (1)$$

Table 1. Chromosome's Fitness Calculation

	Path	Node	Starting Time	Travel Time	Arrive Time	Earliest Time	Waiting Time	Start	Visit Duration	Finish	Latest Time	Penalty
Chromosome	1	4,8	5:00	2:06	7:06	5:00	0:00	7:06	3:00	10:06	19:00	0:00
		8,17	10:06	1:12	11:18	8:00	0:00	11:18	3:00	14:18	17:00	0:00
		17,1	14:18	1:32	15:50	5:00	0:00	15:50	3:00	18:50	17:00	1:50
		1,4	18:50	2:21	21:11	5:00	0:00	21:11	0:00	21:11	19:00	2:11
	2	4,12	1:00	0:37	1:37	5:00	3:23	5:00	3:00	8:00	19:00	0:00
		12,2	8:00	2:02	10:02	5:00	0:00	10:02	3:00	13:02	19:00	0:00
		2,3	13:02	1:02	14:04	5:00	0:00	14:04	3:00	17:04	17:00	0:04
		3,4	17:04	2:57	20:01	5:00	0:00	20:01	0:00	20:01	19:00	1:01
	3	4,14	5:00	0:40	5:40	8:00	2:20	8:00	3:00	11:00	19:00	0:00
		14,10	11:00	2:19	13:19	5:00	0:00	13:19	3:00	16:19	19:00	0:00
		10,4	16:19	2:17	18:36	5:00	0:00	18:36	0:00	18:36	19:00	0:00
		Total		19:05								5:06
		fitness										-4.2101

After getting the result of travel time and penalty, fitness is calculated by using Equation 2.

$$fitness = \frac{1}{1+19.05} + (5.06 \ x - 1) + \frac{8}{10} \qquad (2)$$

$$fitness = -4.2101$$

4.12. Crossover

In this study, crossover process uses one-cut point crossover [22]. The number of offspring obtained from *crossover rate x popsize*. In this method, the cut point value will be generated randomly with a range of values 0 to chromosome length-1.

4.1.3. Mutation

Mutation process using Reciprocal Exchange Mutation [22]. The number of offspring mutations results from *mutation rate x popsize*. In the process of mutation the first step is to select 1 individual at random. In the Reciprocal Exchange Mutation method, 2 randomly selected indexes will be selected with a range of 0-chromosome length-1 and exchange both values of the selected gene index.

4.1.4. Selection

At the selection step, the method to be used is Replacement Selection because can getting maximum solution [1]. The principle of the replacement selection operator is that if the offspring reproduction has a fitness value greater than the parent then offspring will replace the parent Offspring obtained from the mutation process will replace the parent if the offspring has greater fitness than the parent, if the crossover process that has 2 parents, then selected 1 parent with the lowest fitness value [23], [24].

4.2. Simulated Annealing

Simulated annealing (SA) consists of optimizing the "melting" process with the appropriate temperature, then lowering the temperature slowly until it "freezes" and no further process [25]. At the beginning of the process, the metal will be heated. Then, the temperature is lowered slowly to avoid cracking to minimize the energy used [26]. SA's flow can helps GA's individual to get convergence to a minimum global [27]. The higher the temperature and the lower the value of the solution, the greater the chance of receiving a less than optimal solution [28]. Figure 3 shows procedure of simulated annealing which is used.

```
Initialization parameter SA
SA (stopping condition) {
     If (t ≥ final temperature {
          d₁=fitness (chromosome)
          Sₙ=Creating Neighborhood Solution using Scramble Mutation
          d₂=fitness (Sₙ)
          If (d < d₁) {
               Initial α=random [0, 1]
               Initial p=Probability
               If (α < p) {
                    S=Sₙ
               }
          } else {
               S=Sₙ
          }
     }
}
```

Figure 2. Procudure of simulated annealing

Where *S* is maximum chromosome which is get from result of GA. d_1 and d_2 are fitness of *S* and S_n. S_n is new chromosome from Neighborhood scramble mutation process and result solution of SA is *S*. Probabilty value in SA is generated using Equation 3.

$$prob = \frac{1}{e^{\frac{(d_2-d_1)k}{d_1 t}}} \quad (3)$$

4.2.1. Neighborhood Search

In this study, SA will generate neihborhood solution by using Scramble Mutation [27]. At this operator two randomly selected points are selected and switch the chromosome position between two points randomly for example show in Figure 4.
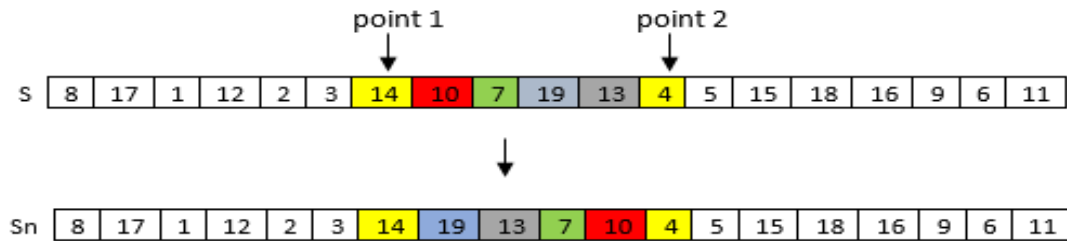
point 1   point 2

S | 8 | 17 | 1 | 12 | 2 | 3 | 14 | 10 | 7 | 19 | 13 | 4 | 5 | 15 | 18 | 16 | 9 | 6 | 11 |

Sn | 8 | 17 | 1 | 12 | 2 | 3 | 14 | 19 | 13 | 7 | 10 | 4 | 5 | 15 | 18 | 16 | 9 | 6 | 11 |

Figure 3. Scramble mutation process

4.3. Hybrid Genetic Algorithm and Simulated Annealing

Based on goodness and weakness each algorithm, combination genetic algorithm and simulated annealing will furnish weakness each algorithm. However, needing one combination to get maximum result. GA-SA1 is prosesed method in this study, while GA-SA2, GA-SA3, and GA-SA4 are comparison method [15].

4.3.1. GA-SA1

Combination GA-SA1 is started by initialization GA parameter. The second process is running SA with initial solution using chromosome with best dan worst fitness. It is stop until iteration maximum in GA. Pseudocode of GA-SA1 can be seen in Figure 5.

```
Initialization parameter GA-SA
GA (stopping condition) {
     //running GA process
     S=chromosome with best fitness of GA
     SA (stopping condition) {
          S=chromosome with best fitness of GA
          //running SA process
     }
     S=chromosome with worst fitness of GA
     SA (stopping condition) {
          S=chromosome with best fitness of GA
          //running SA process
     }
}
```

Figure 4. GA-SA1 process

4.3.2. GA-SA2

GA-SA2 process starts by running GA first until stopping condition, then SA process is executed [15]. In SA process, the number of populations used is obtained based on random rate of the best individuals. Pseudocode of GA-SA2 can be seen in Figure 6.

```
Initialization parameter GA-SA
GA (stopping condition) {
     //running GA process
}
Chromosome=select (random rate, GA chromosome)
 SA (stopping condition) {
 //running SA process
 }
```

Figure 5. GA-SA2 process

4.3.3. GA-SA3

GA-SA3 process starts by running SA algorithm first until stopping condition, then GA process is executed [15]. In GA process, the number of populations used is obtained based on random rate of the best individuals. Pseudocode of GA-SA3 can be seen in Figure 7.

```
Initialization parameter GA-SA
 SA (stopping condition) {
 //running SA process
 }
Chromosome=select (random rate, GA chromosome)
GA (stopping condition) {
      //running GA process
 }
```

Figure 6. GA-SA3 process

4.3.4. GA-SA4

GA-SA2 process starts by running GA first for generations, then SA process is executed and after SA process is completed it will run GA again. After GA process on the first iteration is completed, individual is taken as much as random rate of the best individual to be processed by SA. After SA process is completed, it will take the individual as much as the random rate of the best individual from the SA process to be processed back by GA [15]. The recurrence will stop after stopping condition is found. Pseudocode of GA-SA4 can be seen in Figure 8.

```
Initialization parameter GA-SA
While (Stopping Condition) {
     If (GA condition) {
        GA() {
              //running GA process
        }
     }
     If (SA condition) {
        SA() {
              //running SA process
        }
     }
     Chromosome=select (randomrate, chromosome)
 }
```

Figure 7. GA-SA4 process

5. EXPERIMENTAL RESULT AND DISCUSSION
5.1. Experimental Scenario

The experimental scenario will be done by comparing fitness results generated from each algorithm ie GA, SA, GA-SA1, GA-SA2, GA-SA3, and GA-SA4. In the GA-SA2, GA-SA3, and GA-SA4 algorithms, a random parameter of 0.1-1 is used to select the number of individuals randomly to be processed further. In GA-SA2, if the random rate used is 0.1 then only 10% of the overall GA population will be re-optimized by SA [15]. To know the most optimal random rate, it will be tested random rate parameters in advance so it can be known random rate value that can produce the most optimal fitness value. The parameters used in this experiment can be seen in Table 2.

Table 2. Ecperiment Parameters

	Parameters	Value
	Population Size	400
	Maximum Generation	800
GA	crossover rate	0.3
	mutation rate	0.1

Table 2. Ecperiment Parameters

	Parameters	Value
SA	initial temperature	0.9
	cooling rate	0.9
	new solution acceptance probability coefficient	200

The experiments will be carried out with the same environment using Java programming language and computation time for each algorithm for 60 seconds and will be repeated 10 times to produce a fair comparison.

5.2. Result and Discussion

The first experiment is conducted to determine the optimal random rate. The result of random rate parameter test can be seen in Table 3-5. Table 3 shows that random rate 0.8 yields the highest average fitness value of 1.0323. In Table 4, the highest fitness average is obtained when the random rate is 1 which is 1.0872. In Table 5, the highest average fitness value is obtained when the random rate is 0.5 which is 1.0886.

Table 3. GA-SA2's Random Rate Result

random rate	Fitness Value Iteration										Fitness Average
	1	2	3	4	5	6	7	8	9	10	
0.1	1.0894	1.0067	1.0085	1.0064	1.0085	1.0093	1.0064	1.0101	1.0049	1.0085	1.0159
0.2	1.0094	1.0094	0.9985	1.0059	1.0812	0.9988	1.0093	0.9988	1.0093	1.0891	1.0210
0.3	1.0085	1.0891	1.0064	1.0074	1.0050	1.0093	1.0096	1.0093	1.0812	1.0085	1.0234
0.4	1.0093	1.0059	1.0096	1.0093	0.9956	1.0060	1.0096	1.0897	1.0897	1.0093	1.0234
0.5	1.0093	1.0094	1.0053	1.0049	1.0093	1.0052	1.0101	1.0101	1.0064	1.0101	1.0080
0.6	1.0096	1.0060	1.0053	1.0093	1.0080	1.0101	1.0093	1.0101	0.9982	1.0059	1.0072
0.7	1.0101	1.0101	1.0054	1.0085	1.0094	1.0094	1.0054	1.0074	1.0096	1.0080	1.0083
0.8	**1.0094**	**1.0074**	**1.0064**	**1.0868**	**1.0096**	**1.0891**	**1.0096**	**1.0067**	**1.0883**	**1.0094**	**1.0323**
0.9	1.0074	1.0085	1.0900	1.0101	1.0093	1.0101	1.0094	1.0052	1.0101	1.0064	1.0167
1	0.9985	1.0094	1.0094	1.0053	1.0052	1.0050	1.0054	1.0101	1.0880	1.0085	1.0145

Table 4. GA-SA3's Random Rate Result

random rate	Fitness Value Iteration										Fitness Average
	1	2	3	4	5	6	7	8	9	10	
0.1	0.9991	0.9959	0.9986	1.0047	0.9885	0.7662	1.0094	0.9973	1.0057	0.9984	0.9764
0.2	0.8816	1.0093	0.9978	0.9983	1.0085	1.0827	0.7615	0.8551	0.8573	1.0888	0.9541
0.3	1.0053	0.8803	0.8182	0.8696	0.8694	0.8571	1.0045	0.8613	0.7623	1.0062	0.8934
0.4	1.0891	0.8000	0.8648	0.6706	1.0067	0.8609	0.9965	1.0064	1.0096	0.8614	0.9166
0.5	0.7833	0.7739	1.0054	0.8644	1.0046	0.8658	0.8617	1.0057	1.0043	0.8583	0.9027
0.6	1.0864	1.0067	0.7800	1.0053	0.7662	0.7876	0.7694	1.0076	0.8688	1.0093	0.9087
0.7	0.8899	0.8664	0.8402	1.0080	0.8796	0.8810	1.0064	1.0052	0.8554	0.8702	0.9102
0.8	1.0060	0.8649	1.0064	0.9153	0.8655	0.8751	1.0082	0.8543	0.8699	0.8661	0.9132
0.9	0.8694	0.9570	1.0049	0.9654	0.8619	0.9993	0.9604	0.8666	0.8816	0.8700	0.9237
1	**1.0888**	**1.0867**	**1.0900**	**1.0888**	**1.0867**	**1.0868**	**1.0866**	**1.0867**	**1.0888**	**1.0822**	**1.0872**

Table 5. GA-SA3's Random Rate Result

random rate	Fitness Value Iteration										Fitness Average
	1	2	3	4	5	6	7	8	9	10	
0.1	1.09	1.0812	1.0812	1.0868	1.0888	1.0888	1.0867	1.0888	1.0883	1.0888	1.0869
0.2	1.0867	1.0868	1.0868	1.0888	1.09	1.0883	1.0888	1.0826	1.0888	1.09	1.0878
0.3	1.0884	1.09	1.0888	1.0897	1.0867	1.0884	1.0867	1.0891	1.0812	1.0867	1.0876
0.4	1.0868	1.0867	1.0897	1.0868	1.09	1.0867	1.0888	1.0867	1.0868	1.0894	1.0878
0.5	**1.0884**	**1.0897**	**1.0868**	**1.0897**	**1.0867**	**1.0868**	**1.0897**	**1.09**	**1.0894**	**1.0894**	**1.0886**
0.6	1.0867	1.0868	1.0888	1.0888	1.0868	1.0868	1.0864	1.0894	1.0888	1.0897	1.0879
0.7	1.0868	1.0884	1.09	1.0868	1.0864	1.0894	1.0888	1.0894	1.09	1.0867	1.0883
0.8	1.0868	1.0888	1.0894	1.0812	1.0868	1.0894	1.0888	1.0812	1.088	1.0868	1.0867
0.9	1.0868	1.0894	1.0897	1.0868	1.0897	1.0888	1.09	1.0888	1.0868	1.0897	1.0886
1	1.0894	1.09	1.0867	1.0884	1.0868	1.0867	1.0888	1.0867	1.0894	1.0867	1.0879

The next experiment is to compare the fitness results of each algorithm using GA and SA parameters and the random rate parameters that have been tested previously. Testing is done 10 times on each algorithm. The comparison of the algorithm can be seen in Table 6.

The first test was performed on a genetic algorithm. After testing 10 times, the average fitness score of 1.0384 was obtained. The second experiment was performed on the simulated annealing algorithm and produced an average fitness of 1.0772. The third test was performed on the GA-SA1 hybridization and generated an average fitness of 1.0888. In GA-SA2 test, the average fitness gain is 1.0135. In the GA-SA4 test the average fitness gain of 1.0124 and the GA-SA4 test obtained the average fitness value of 1.0863. From the sixth test of the algorithm, the best average fitness value is generated by the GA-SA1.

Figure 9 shows the fitness results obtained by each algorithm within 1 minute. The chart shows that uses of the simulated annealing algorithm can significantly increase the fitness value because e the algorithm effectively exploits local search area, providing an opportunity to find the best fitness faster. From all algorithms tested in this research, GA-SA1 and GA-SA4 algorithm can produce the most optimal fitness among other algorithms, but GA-SA4 can find the best fitness chromosome for 26 second faster than GA-SA1 that requires 39 second to find the best fitness chromosome. however, GA-SA4 is still often stuck on its local search while GA-SA1 can find the optimum point even though it takes a little longer.

Table 6. Result Comparison of Algorithm

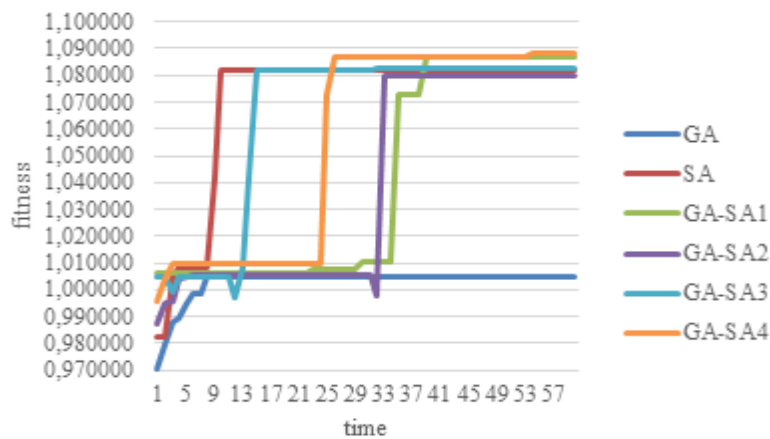| Iteration	Algorithm					
	GA	SA	GA-SA1	GA-SA2	GA-SA3	GA-SA4
1	1.0900	1.0816	1.0884	1.0085	1.0891	1.0867
2	1.0101	1.0876	1.0868	1.0093	1.0866	1.0868
3	1.0888	1.0816	1.0900	1.0064	1.0891	1.0894
4	0.9982	1.0867	1.0894	1.0094	1.0867	1.0900
5	1.0060	1.0868	1.0868	1.0067	1.0867	1.0894
6	1.0094	1.0812	1.0894	1.0057	1.0897	1.0868
7	1.0883	1.0868	1.0894	1.0900	1.0822	1.0897
8	1.0054	1.0064	1.0900	1.0054	1.0900	1.0884
9	1.0891	1.0868	1.0897	1.0047	1.0888	1.0897
10	0.9984	1.0867	1.0884	1.0900	1.0891	1.0897
Fitness Average	1.0384	1.0772	**1.0888**	1.0236	1.0878	1.0887
Fitness Minimum	0.9982	1.0064	1.0868	1.0047	1.0822	1.0867
Fitness Maximum	1.0900	1.0876	1.0900	1.0900	1.0900	1.0900

Figure 8. Comparison Result

6. CONCLUSION

This study compares the fitness values generated by genetic algorithm and simulated annealing algorithm and four its hybridization models. From the sixth test of the algorithm, the best average fitness value is generated by the GA-SA1. However, the ratio of fitness values generated by GA-SA1 and GA-SA4 can be said to be close together so that both GA-SA1 and GA-SA4 can produce the most optimal average

fitness values but GA-SA4 can find the best fitness chromosome faster than GA-SA1. Determination of population number of genetic algorithm to be optimized again by simulated annealing algorithm can influence algorithm speed in finding best result. In GA-SA1, populations re-optimized by the simulated annealing algorithm are just the best and worst fitness chromosomes, but in GA-SA4, the population is 10% of the population with the best chromosomes, the more chromosomes optimized by the simuated annealing algorithm, the faster the chance of the algorithm to find the most optimal fitness. Futher research, to needs K-Means for clustering nodes that can improve computation time [30].

REFERENCES

[1]   Y. P. Anggodo, *et al.,* "Optimation of Multi-Trip Vehicle Routing Problem with Time Windows using Genetic Algorithm," *Journal of Environmental Engineering and Sustainable Technology*, vol. 3, no. 2, pp. 92-97, Nov 2016.

[2]   X, Ji and K. Yao, "Uncertain Project Scheduling Problem with Resource Constraints," *Journal of Intelligent Manufacturing*, vol. 28, issue 3, pp. 575-580, Oct 2014.

[3]   C. Verbeeck, *et al.,* "A Metaheuristic Solution Approach for The Time-Constrained Project Scheduling Problem," *OR Spectrum*, vol 39, issue 2, pp. 353-371, Mar 2016.

[4]   S. Sundar, *et al.,* "A Hybrid Artificial Bee Colony Algorithm for The Job-Shop Scheduling Problem with No-Wait Constraint," *Soft Computing*, vol. 12, issue 5, pp. 1193-1202, Mar 2017.

[5]   K. Ghoseiri and S. F. Ghannadpour., "Multi-Objective Vehicle Routing Problem with Time Windows Using Goal Programming and Genetic Algorithm," *Applied Soft Computing*, vol. 10, issue 4, Sep 2010.

[6]   Y. Bai, et al., *"Study of Multi-Vehicle Routing Problem with Time Window,"* twelfth *International* Symposium on Operations Research and its Applications in engineering, technology and management (ISORA), 2015.

[7]   K. Heng, et al., *"Application of Route Flexibility in Data-Starved Vehicle Routing Problem with Time Windows,".* IEEE Congress on Evolutionary Computation (CEC), 2016.

[8]   R. Chunyu and W. Xiaobo., *"Research on Multi-Vehicle and Multi-Depot Vehicle Routing Problem with Time Windows for Electronic Commere,".* International Conference on Artificial Intelligence and Computational Intelligence, 2010.

[9]   Z. Ursani, *et al.,* "Localized Genetic Algorithm for Vehicle Routing Problem with Time Windows, "*Applied Soft Computing*, vol. 11, issue 8, pp. 5375-5390, Dec 2011.

[10] S. Karakatic and V. Podgorelec., "A Survey of Genetic Algorithms for Solving Multi Depot Vehicle Routing Problem," *Applied Soft Computing*, vol. 27, pp. 519-532, Feb 2015.

[11] S. N. Kumar and R. Panneerselvam., "A Time-Dependent Vehicle Routing Problem with Time Windows for E-Commerce Supplier Site Pickups Using Genetic Algorithm," *Intelligent Information Management*, vol. 7, no. 4, pp. 181-194, Jul 2015.

[12] Y. Wang, *et al.,* "Two-Echelon Logistics Distribution Region Partitioning Problem Based on Hybrid Swarm Optimization-Genetic Algorithm," *Expert Systems with Applications*, vol 42, issue 12, Jul 2015.

[13] D. M. Pierre and N. Zakaria., "Stochastic Partially Optimized Cyclic Shift Crossover for Multi-Objective Genetic Algorithms for The Vehicle Routing Problem with Time-Windows," *Applied Soft Computing*, vol. 52, pp. 863-876, Mar 2017.

[14] W. F. Mahmudy., "Improved Simulated Annealing for Optimization of Vehicle Routing Problem with Time Windows (VRPTW)," *Kursor*, vol. 7, no. 3, pp. 109-116, 2017.

[15] T. N. Fatyanosa, *et al.,* "Hybrid Genetic Algorithm and Simulated Annealing for Function Optimization," *Journal of Information Technology and Computer Science*. Vol. 1, No. 2, pp. 82-97.1

[16] O. Bahri., "Multi-Objective Optimiation of Vehicle Routing Problem with uncertainty: An Approach Based on Possibility Theory," 2012. < http://www.larodec.com/download/file/fid/543> [Access on January 25 2017]

[17] S. W. Lin, *et al.* "Vehicle Routing Problems with Time Windows Using Simulated Annealing," *Systems, Man, and Cybernetics*, pp. 645-650, Jul 2017.

[18] N. Wassan, et al. "The Multiple Trip Vehicle Routing Problem with Backhauls: Formulation and a Two-Level Variable Neighborhood Search," *Computer & Operations Research*, vol. 78, pp. 454-467, Feb 2017.

[19] Y. Yang and L. Tang., *"A Filter-and-fan Approach to the Multi-Trip Vehicle Routing Problem,"* International Conference on Logistics Systems and Intelligent Management, vol. 3, pp. 1713-1717, Jan 2010.

[20] Y. Y. Zhang, et al., *"Optimal Reservoir Operation Using a Hybrid Simulated Annealing Algorithm-Genetic Algorithm,"* International Conference on Bio-Inspired Computing: Theories and Applications, pp. 454-458, Nov 2010.

[21] B. Minocha, et al., *"Solving Vehicle Routing and Scheduling Problems using Hybrid Genetic Algorithm".* International Conference on Electronics Computer Technology, 2011, pp. 189-193.

[22] V. N. Wijayaningrum and W. F. Mahmudy., "Optimization of Ship's Route Scheduling Using Genetic Algorithm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS),* vol. 2, no. 1, pp. 180-186, Apr 2016.

[23] W. F. Mahmudy, *et al.,* "Real Coded Genetic Algorithms for Solving Flexible Job-Shop Scheduling Problem – Part I: Modeling,", *Advanced Materials Research,* vol. 701, pp. 359-363, May 2013.

[24] W. F. Mahmudy, *et al.,* "Real Coded Genetic Algorithms for Solving Flexible Job-Shop Scheduling Problem – Part II: Optimization," *Advanced Materials Research,* vol. 701, pp. 364-369, May 2013.

[25] S. Kickpatrick, *et al.,* "Optimization by Simulated Annealing," *Science*, vol. 220, issue 4598, pp. 671-680, May 1983.

[26] K. T. Tung, *et al.,* "Solving Cutting Scheduling Problem by Simulated Annealing Search Method," *International Conference on Industrial Engineering and Engineering Management*, 2016, pp. 907-911.

[27] M. Lalaoui, *et al.,* "A Self-Tuned Simulated Annealing Algorithm using Hiddeng Markov Model," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 8, no. 1, pp. 291-298, Feb 2018.

[28] J. B. Chagas, *et al.,* "Simulated Annealing Metaheuristic for the Double Vehicle Routing Problem with Multiple Stacks," *International Conference on Intelligent Transportation Systems*, 2016, pp. 1311-1316.

[29] H. C. B. Oliveira, *et al.,* "A Robust Method for the VRPTW with Multi-Start Simulated Annealing and Statistical Analysis," *Symposium on Computational Intelligence in Scheduling*, 2007, pp 198-205.

[30] A. N. Alfiyatin, *et al*., "K-Means and Genetic Algorithm to Solve Vehicle Routing Problem with Time Windows Problem," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS),* vol. 11, no. 2, pp. 464-468, Aug 2018.